知的情報処理論（第6回）

2023年5月23日（火）
産業科学研究所
駒谷 和範

本日（第6回）の内容

- 第1回　人工知能の歴史
- 第2回第3回第4回　機械学習の基礎
- 第5回　対話システムの構成，言語理解
- 第6回
  - （大規模）言語モデル
- 第7回
  - 言語理解の利用法
  - 対話管理，ニューラル生成，言語以外の要素
- 第8回　人工知能と社会

言語理解（残り）

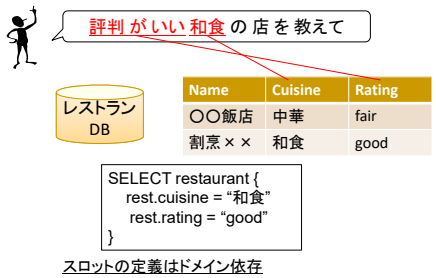
言語理解部の中身

理解結果（中間表現：フレーム表現）を出力

① ドメイン　発話ごと
② 意図　発話ごと
③ スロット値　単語列ごと

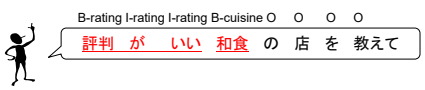
⇒ 後段の対話管理部に渡される（いわゆる「対話型」）

3. スロット値同定



スロットの定義はドメイン依存

機械学習によるスロット値同定

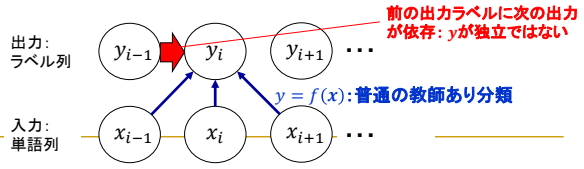


- 系列ラベリング問題
  - 系列内の各要素にラベルをつける
  - 各ラベルは系列の前後の要素に依存
- IOB2ラベリング
  - ラベルが複数語からなる場合を表現
  - Bは開始点(beginning)，Iはその内部(inside)，Oは外部(outside)

スロット値同定

週末の大阪の天気を教えて
[Date] [City]

- キーワードの抽出
  - 単純手法：辞書中の単語とマッチング
  - 単純手法では誤りとなる例：「大阪駅のそばにある」
- 単語文脈を考慮した機械学習：系列ラベリング問題



機械学習による系列ラベリング

- 入力：単語の系列
- 出力：IOB2ラベルの系列
  - 単語列のどの部分がスロット値かがわかる

系列ラベリング問題

- 分類器
  - Conditional Random Field (CRF), Hidden Markov Model (HMM), …
  - ニューラルネットワーク（RNN, LSTM）

9

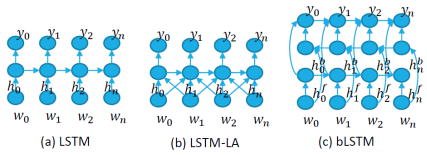
固有表現抽出

- 自然言語処理のタスクとしてポピュラーな系列ラベリング問題
- テキストの中から，固有表現（Named Entity: NE）を抽出
  - 組織名，人名，地名，日付など

田中	将大	が	神戸	の	マウンド	に	立つ
B-NAME	I-NAME	O	B-PLACE	O	O	O	O

ニューラルネットワークによるスロット値同定（1）

- RNNの利用 [Mesnil et al., 2013]など
  a. LSTM
  b. n-gramに窓をかけて入力に使用
  c. 双方向LSTM

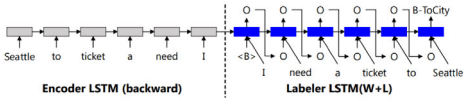


http://deepdialogue.miulab.tw/ より

11

ニューラルネットワークによるスロット値同定（2）

- 機械翻訳で用いられる系列変換モデルを適用
  - Encoder-Decoderネットワーク [Kurata et al., 2016]

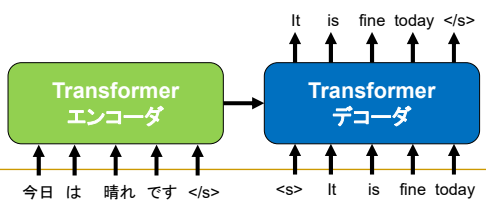


  - 注意機構（attention mechanism）の利用 [Simonnet et al., 2015]
    - 系列のj番目の出力時の，i番目の隠れ状態による重み
    - （例えば）feed-forward NNにより学習

教科書：
深層学習による自然言語処理 (2017)

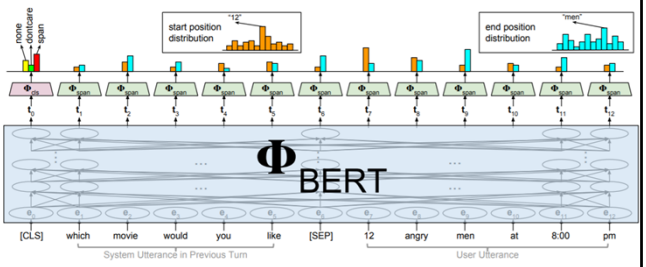
RNN/LSTMはTransformerへ

- Transformer [Vaswani+ (Google), 2017]
  https://arxiv.org/pdf/1706.03762.pdf
  - RNNやCNNを用いず，アテンション（自己注意）を用いたエンコーダデコーダ型モデル
  - 機械翻訳で当時の最高性能を更新



BERTを使ったスロット値同定

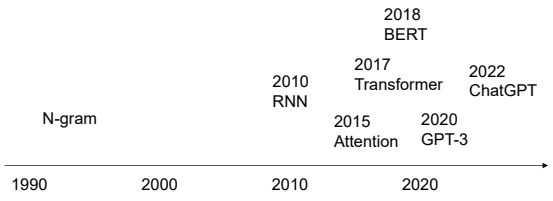
https://arxiv.org/pdf/1907.03040.pdf



- User Utteranceの中で，映画名のspan（始点と終点となる単語）を当てる

言語モデル

目次（言語モデルの歴史）



言語モデル

- 与えられた単語列 $W = w_1 w_2 \ldots w_n$ に対し、その出現確率 $P(W) = P(w_1 w_2 \ldots w_n)$ を与える確率モデル
  - 例：P(今日/は/良/い/天気/です/ね)＝1.158×10^{-115}
    P(は/今日/良/です/天気/い/ね)＝5.275×10^{-337}
- 単位：トークン (token)
  - （基本は）単語
  - 日本語の場合，事前に文から単語への分かち書き（と読みの付与）が必要
    → 形態素解析

- テキストの続き（あるテキストに続く単語）を予測できる

$$y^* = argmax_{\{y \in V\}} P(\text{ドラえもん}, \text{の}, \text{好物}, \text{は}, y)$$

全単語の集合

$P(\text{ドラえもん}, \text{の}, \text{好物}, \text{は}, \text{せんべい}) = 0.00000021$
$P(\text{ドラえもん}, \text{の}, \text{好物}, \text{は}, \text{まんじゅう}) = 0.00000026$
$P(\text{ドラえもん}, \text{の}, \text{好物}, \text{は}, \ldots\ldots) = \cdots$
$P(\text{ドラえもん}, \text{の}, \text{好物}, \text{は}, \text{どら焼き}) = 0.00000113$ y^* =どら焼き

- 様々なタスクは言語モデルでも定式化できる
  - 機械翻訳
    P(The, favorite, of, Doraemon, is, Dorayaki, [SEP], ドラえもん, の, 好物, は, y)

条件付き確率による言語モデル

- 文全体の確率を，部分の条件付確率の積で表現
  - 単語列が長くなるほど，その単語列そのものはあまり生起しない
    → データスパースネス (data sparseness) 問題
    - Webにも完全に一致する単語列はなかなかない

N-gramモデル

- 単純な統計的言語モデル
- ある単語の出現確率が、直前の(N-1)単語のみに依存すると仮定

$$P(w_1 w_2 \cdots w_n) = \prod_{i=1}^{n} P(w_i \mid w_{i-N+1} \cdots w_{i-1})$$

  - $N = 1$: ユニグラム（unigram）
  - $N = 2$: バイグラム（bigram）
  - $N = 3$: トライグラム（trigram）

トライグラムの例

- 「大学/に/行/く」という単語列の出現確率は？

P(<s>, 大学, に, 行, く, </s>)
= P(大学|<s>)
× P(に|<s>, 大学)
× P(行|大学, に)
× P(く|に, 行)
× P(</s>|行, く)

ただし<s>, </s>はそれぞれ文頭，文末を表す

N-gram確率の算出（学習）

- 最尤推定法によるNグラム確率の算出（トライグラムの場合）

$$P(w_i|w_{i-2}w_{i-1}) = \frac{N(w_{i-2}w_{i-1}w_i)}{N(w_{i-2}w_{i-1})}$$

$N(w_{i-2}w_{i-1}w_i)$: $w_{i-2}w_{i-1}w_i$の出現頻度

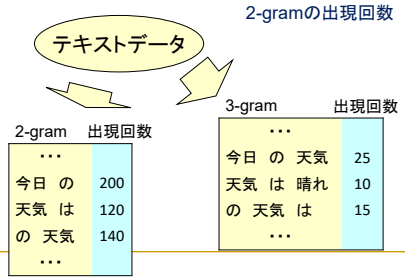
最尤推定 (Maximum likelihood estimation):
与えられたデータに対して，モデルの尤度が最大となるようにパラメータを決める方法

N-gram確率算出の例

3-gramの出現回数

$$P(\text{晴れ}|\text{天気 は}) = \frac{N(\text{天気 は 晴れ})}{N(\text{天気 は})}$$

2-gramの出現回数



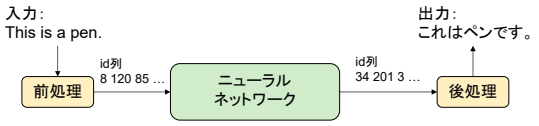
N-gramモデル

- 音声認識の言語モデルとして長い間利用されてきた
  - 入力された音声波形に対する，単語の並びの可能性（探索空間）を狭めるのに有用
- 問題点
  - ① 長距離依存（N-1単語を超えた文脈）を扱いにくい
    トライグラム（N=3）の例:
    $P(y \mid \text{ドラえもん}, \text{の}, \text{好物}, \text{は}) \approx P(y \mid \text{好物}, \text{は})$
    $= \frac{\#(\text{好物}, \text{は}, y)}{\#(\text{好物}, \text{は})}$ 局所的な文脈のみからの予測になってしまう
  - ② 類義語問題
    - トークンが違うと全く別の語彙
      例：天気 vs 天候
      別個のトークンとしてそれぞれ学習する必要

ニューラルモデルでの単語のベクトル表現

ことばに対する深層学習

- 文を、トークンID（基本は単語単位）の列として扱う
- それらの間での処理（機械翻訳での例）：



  - 前処理，後処理：単語とID列の変換 (tokenize)
    - 日本語の場合は単語分割（形態素解析）必要
    - 「単語」の定義は、一意には定まらない（例：大阪大学）
    - サブワード，sentencepiece, …
    - IDが付与されていない文字列＝未知語
  - 単語の指す内容を把握して処理しているわけではない

One-hotベクトル表現

- 各単語IDに1次元を割り当てる
  - 語彙サイズ＝次元数

$$\begin{array}{l} \text{apple} \\ \text{banana} \\ \text{dog} \\ \text{cat} \end{array} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

  - 各単語がすべて直交（内積=0）
  - ニューラルネットワークの入力にするには次元が大きすぎる
    - 数万語～数百万語

分散表現

- 元々の定義：ある単語の意味を、周囲の文脈（単語の共起）で表現
- 深層学習では
  - 単語をより低次元（100～1000次元）のベクトルで表現
    - ニューラルネットワークの入力に耐えうる
  - ベクトル（行列）の中身も学習対象
    - 単語間の類似度も反映 cf. Word2vec
    - cf. 主成分分析（Principal Component Analysis, PCA）

②類義語問題は（ある程度）解決

$$\begin{array}{l} \text{apple} \\ \text{banana} \\ \text{dog} \\ \text{cat} \end{array} \begin{bmatrix} 1.0 & 0.3 & -0.5 \\ 0.8 & -0.2 & 0.1 \\ 0.2 & 0.7 & -0.2 \\ 0.1 & 0.8 & -0.3 \end{bmatrix}$$

入力：埋め込み（embedding）レイヤ

$$\begin{array}{l} \text{apple} \\ \text{banana} \\ \text{dog} \\ \text{cat} \end{array} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1.0 & 0.3 & -0.5 \\ 0.8 & -0.2 & 0.1 \\ 0.2 & 0.7 & -0.2 \\ 0.1 & 0.8 & -0.3 \end{bmatrix}$$

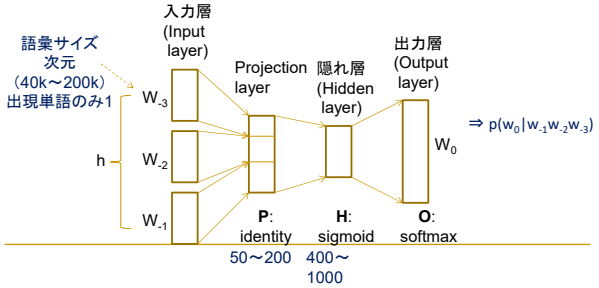
入力単語

$$[0\ 1\ 0\ 0] \cdot \begin{bmatrix} 1.0 & 0.3 & -0.5 \\ 0.8 & -0.2 & 0.1 \\ 0.2 & 0.7 & -0.2 \\ 0.1 & 0.8 & -0.3 \end{bmatrix} = [0.8\ \ -0.2\ \ 0.1]$$

bananaの行のみを取り出すことが、行列の演算のみで可能

ニューラルネットワークに基づく言語モデル

- フィードフォワード型
  - 入力（＝過去N単語の履歴）から次単語を予測
  - 入力の表現：Vector Space Model (One-hot representation)



出力：全結合→ソフトマックス

ネットワークからの出力

$$\begin{array}{l} \text{apple} \\ \text{banana} \\ \text{dog} \\ \text{cat} \end{array} \begin{bmatrix} 1.0 & 0.3 & -0.5 \\ 0.8 & -0.2 & 0.1 \\ 0.2 & 0.7 & -0.2 \\ 0.1 & 0.8 & -0.3 \end{bmatrix} \cdot \begin{bmatrix} 0.3 \\ 0.1 \\ -0.2 \end{bmatrix} = \begin{bmatrix} 0.34 \\ 0.2 \\ 0.17 \\ 0.17 \end{bmatrix}$$

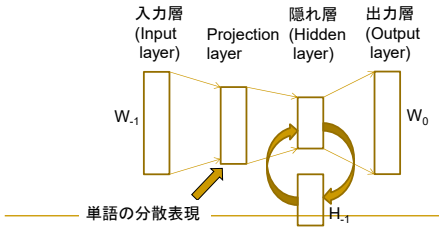
$$\xrightarrow{\text{softmax}} \begin{bmatrix} 0.281 \\ 0.244 \\ 0.237 \\ 0.237 \end{bmatrix} \xrightarrow{\text{argmax}} \text{ID: 0 (=apple)}$$

ネットワークから出力されたベクトルをもとに、最も尤もらしい単語を出力するなどできる

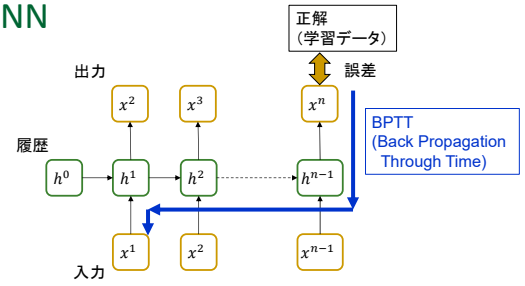
ニューラルネットワークに基づく言語モデル

- リカレント型（RNN）
  - 1つ前の隠れ層の出力をフィードバック
  - 原理的には全ての履歴を考慮

①長距離依存を（ある程度）表現



RNN



- 文が長くなると単語位置方向にネットワークが深くなるため，学習が難しい（勾配消失問題）
  - LSTM，GRU
- 並列化が困難
- 固定長のベクトルでは情報を覚えきれない

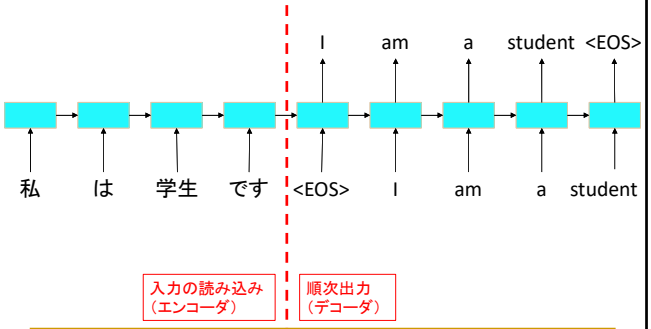
Mini Quiz #1

- 文内の離れた位置にある単語間の関係（制約）をRNNはどのように表そうとしているか。またそれが実際の学習では必ずしもうまく行かない理由を考えよ。

系列変換モデル・注意機構 (ATTENTION)

ニューラル機械翻訳（系列変換モデル）

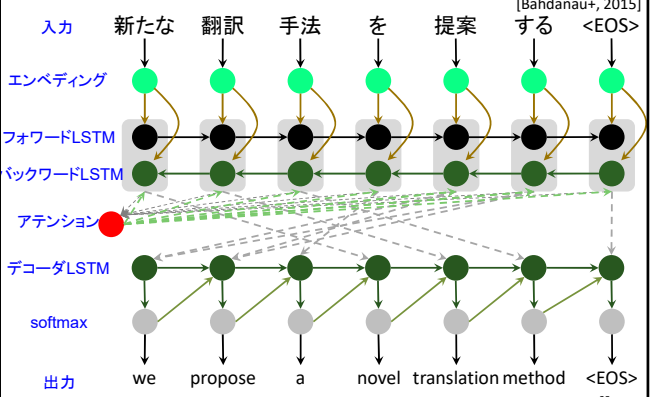
[Sutskever+ 2014]



37

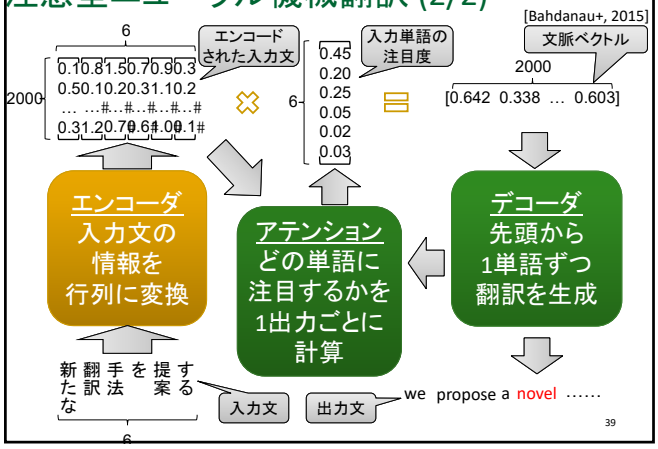
注意型ニューラル機械翻訳 (1/2)

[Bahdanau+, 2015]



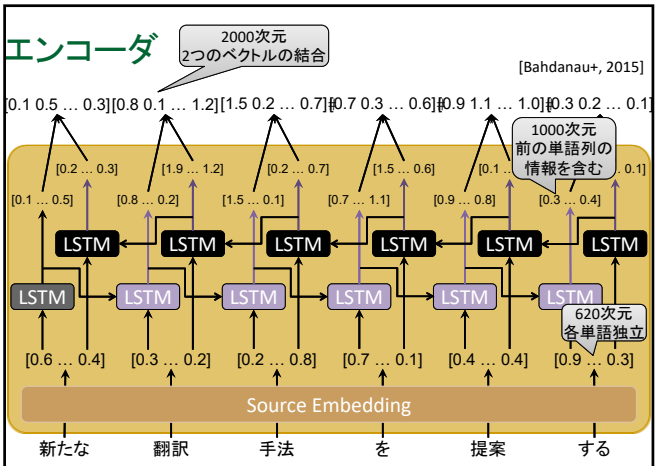
38

注意型ニューラル機械翻訳 (2/2)



39

エンコーダ

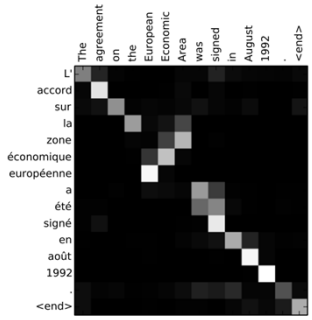


40

アテンションの例

[Bahdanau+, 2015]
https://arxiv.org/pdf/1409.0473.pdf

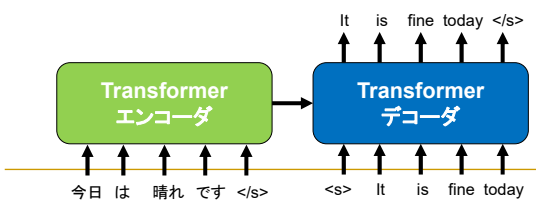
英語からフランス語への翻訳の際の例



TRANSFORMER （自己注意型）

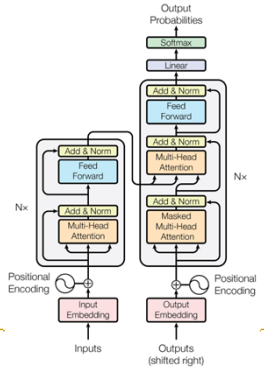
RNN/LSTMはTransformerへ

- Transformer [Vaswani+ (Google), 2017]
  https://arxiv.org/pdf/1706.03762.pdf
  - RNNやCNNを用いず，アテンション（自己注意）のみを用いたエンコーダデコーダ型モデル
  - 機械翻訳で当時の最高性能を更新



自己注意型 ニューラル機械翻訳 (Transformer)

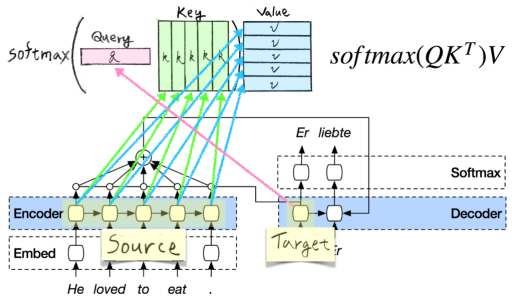
[Vaswani+, 2017]



6段ずつ

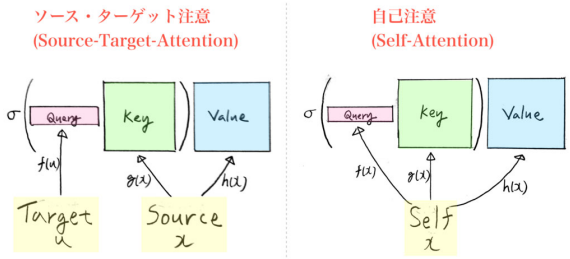
45

注意機構におけるQuery, Key, Value



図の出典：
http://deeplearning.hatenablog.com/entry/transformer

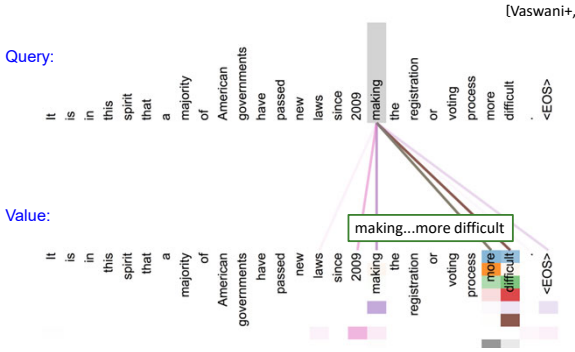
ソースターゲット注意と自己注意



図の出典：
http://deeplearning.hatenablog.com/entry/transformer

自己注意の可視化 (1/2)

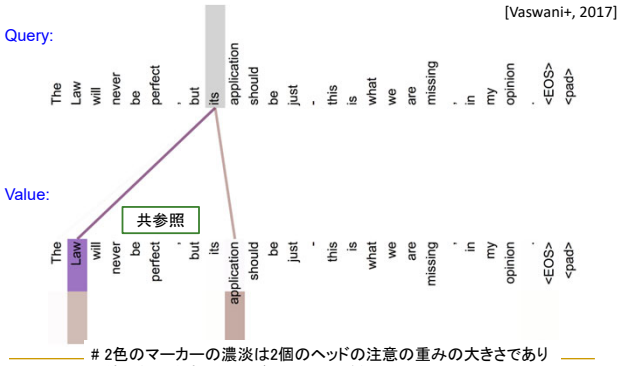
[Vaswani+, 2017]



8色のマーカーの濃淡は8個のヘッドの注意の重みの大きさであり
線の色は注意の重みが最大のヘッドを示している

自己注意の可視化 (2/2)

[Vaswani+, 2017]



2色のマーカーの濃淡は2個のヘッドの注意の重みの大きさであり
線の色は注意の重みが最大のヘッドを示している

Mini Quiz #2

- Transformerは，RNN型系列変換モデルの問題をどのような仕掛けにより解消したか確認せよ。
  - 学習の並列化が困難
  - 勾配消失問題（長距離の依存関係を覚えられない）

事前学習済み言語モデル

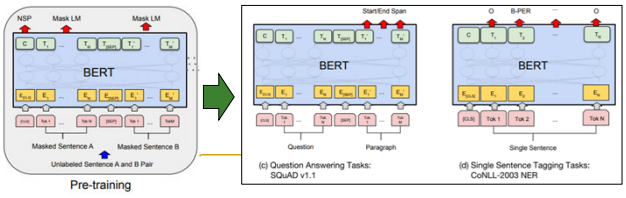
ベースとなるモデルTransformer
- 入力はベクトル（＝単語に限らない；画像でもOK）

- BERT (Google)：エンコーダ
  - テキスト（単語系列）のクラス分類，各単語へのラベリング
- GPT-3 (OpenAI): デコーダ
  - 入力系列の続きを予測
  - （自然言語）生成
  - https://arxiv.org/abs/2005.14165
- T5 (Google)：デコーダ
  - テキストAからテキストBに変換（翻訳や要約など）
  - https://arxiv.org/abs/1910.10683

これらをファインチューニングして使うのが2020年前後の定石

BERT (2018.10) http://arxiv.org/abs/1810.04805
https://www.aclweb.org/anthology/N19-1423

- 事前学習済みの24層Transformerエンコーダ
  - 文を時系列ではなく一括で入力
    - セルフアテンション (self-attention)
  - 事前学習＋(少量データで)ファインチューニング
  - 「汎用言語モデル」



GPT-3

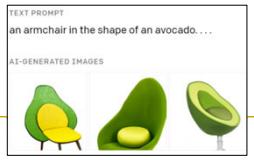
- Generative Pre-trained Transformer 3
  - 言語モデル：1750億パラメータ
- Few-shot learning（少数の例からの学習）
  - プロンプトプログラミング

English: Where is a good restaurant?
Japanese: どこかに良いレストランはありますか?
English: Where is the restroom?
Japanese:

プロンプト
（GPT-3への入力）

→トイレはどこですか？

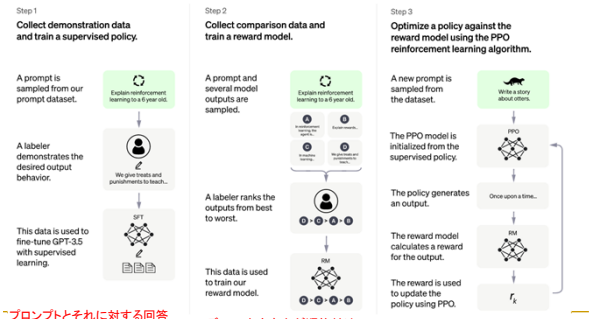
- Dall-eもGPT-3
  - https://openai.com/blog/dall-e/



CHATGPT

ChatGPTのRLHF
https://openai.com/blog/chatgpt

(Reinforcement Learning from Human Feedback)

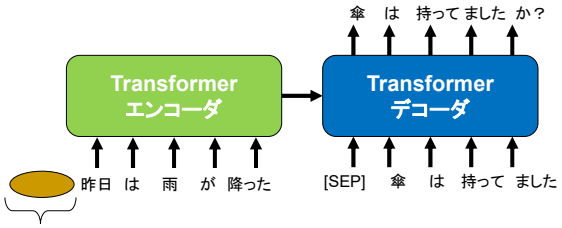


プロンプトとそれに対する回答の組を人手で作成
これでファインチューニング

モデルの出力を人が順位付け
順位付け（報酬）モデルを学習

報酬モデルを使ってGPTモデルを強化学習

プロンプトエンジニアリング



実際の入力の前に，指示 (zero-shot) や例 (few-shot)を書いてもよい

- ファインチューニングをしなくても動く
- ChatGPTはブラウザ上でいろいろ試せる
  - https://www.promptingguide.ai/jp

ChatGPT

- 2023年春現在で社会に大きなインパクト
  - 要素技術は成熟しつつあった
  - Webブラウザで{Zero, Few}-shot で動くのが大きい
    → ファインチューニングなど考えなくてもある程度の性能を達成
- 基本はTransformer（デコーダ）
- 学習に用いたデータの著作権問題