

# データサイエンス 『実践コース』

## 数理工学PBL

---

Day 2-3：大規模言語モデル

東北大学 小池 敦

# 大規模言語モデル の応用（生成AI）

---

# 主な会話AIサービス

---

- ChatGPT
  - OpenAIが提供
  - 各種APIも公開
  - GPT Builderにより，自分用GPTを作れる
- Gemini
  - Googleが提供
  - GPT4と同レベル？
- Claude
  - Anthropic社というスタートアップが提供
  - 長い文書の読解能力が高い印象
  - Claude3が高性能と話題

# 会話AIサービスの高性能化

---

- プロンプトエンジニアリング
  - AIへの指示を工夫することで適切な回答を得る
    - 答え方の具体例を示すことで所望のフォーマットで回答してもらう (Few-shot prompting)
    - Step-by-stepで回答してもらうことで間違いにくくなる
    - 曖昧さの少ない指示をする
- ファインチューニング
  - 自分のデータを使ってモデルを再学習することでより自分好みの回答が得られるようにする
- RAG
  - 外部の情報 (Web検索結果等) を参照しながら回答することで未知の情報への対応や、正確な回答を行う

# 大規模言語モデルを活用した画像生成

---

- 大規模言語モデルが理解した内容を用いて画像を生成する
- 主なサービス
  - DALL・E 3 (OpenAI)  
<https://labs.openai.com>
  - Stable Diffusion (ミュンヘン大学；ソース公開)
    - Stable Diffusion Online: <https://stablediffusion.fr>
  - Midjourney (Midjourney社)  
<https://midjourney.com/> (Discord上で生成してもらう)
  - Adobe Firefly

# ChatGPTによる対話例



プロジェクトリーダーになりました。何を勉強するべきですか？



プロジェクトリーダーになることは素晴らしいことです。以下がプロジェクトリーダーになる上で勉強すべきトピックです：

1. プロジェクトマネジメントのフレームワーク: PMBOK、Agile、Scrumなどのフレームワークを学ぶことが重要です。
2. コミュニケーションスキル: チームメンバー、ステークホルダー、上司とのコミュニケーションスキルを向上させることが大切です。
3. リーダーシップスキル: チームのモチベーションを向上させ、達成感を与えるためのリーダーシップスキルを身に付けましょう。
4. リスクマネジメント: プロジェクトのリスクを認識し、対応するためのスキルを向上させましょう。
5. プロジェクトスケジューリングとバジェット管理: 時間と資金を効果的に管理するためのスキルを向上させましょう。

これらはプロジェクトリーダーになる上で重要なトピックですが、他にも多くのトピックがあります。常に学習を続けて、スキルを向上させ続けることが大切です。



加速度データを解析するKerasモデルのコードを知りたいです。



Kerasを使った加速度データの解析モデルのコードを以下に示します。このコードは、加速度データを入力として、それが人間からのアクティビティか、それ以外のものかを分類するようなモデルを構築するものです。

Copy code

```
# ライブラリのインポート
import numpy as np
import keras
from keras.models import Sequential
from keras.layers import Dense, Dropout

# 加速度データとラベルを読み込む
X_train = np.load('acceleration_data.npy')
y_train = np.load('labels.npy')

# 入力データの次元数を取得する
input_shape = X_train.shape[1]

# モデルを構築する
model = Sequential()
model.add(Dense(64, activation='relu', input_shape=(input_shape,)))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(1, activation='sigmoid'))

# モデルをコンパイルする
model.compile(loss='binary_crossentropy',
              optimizer='adam',
              metrics=['accuracy'])

# モデルを学習させる
model.fit(X_train, y_train, epochs=10, batch_size=32)

# モデルを評価する
scores = model.evaluate(X_test, y_test, verbose=1)
print('Test loss:', scores[0])
print('Test accuracy:', scores[1])

# モデルを使って予測を行う
predictions = model.predict(X_test)
```

# ChatGPTによるプログラミング



X\_testが見つかりません



お詫びします。先ほどのコードでは、X\_testとy\_testが定義されていませんでした。これらの変数は、学習用のデータとテスト用のデータを分割して作成する必要があります。

例えば、以下のように、データを学習用のデータとテスト用のデータに分割することができます。

Copy code

```
# データを学習用とテスト用に分割する
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.2)
```

上記のコードでは、Xとyを学習用データとテスト用データに分割しています。X\_trainには学習用のデータが、X\_testにはテスト用のデータが、y\_trainには学習用のラベルが、y\_testにはテスト用のラベルが格納されます。

ご参考にしていただければ幸いです。

間違えることもあるが、  
指摘するときちゃんと修正する

# ChatGPTのインパクト

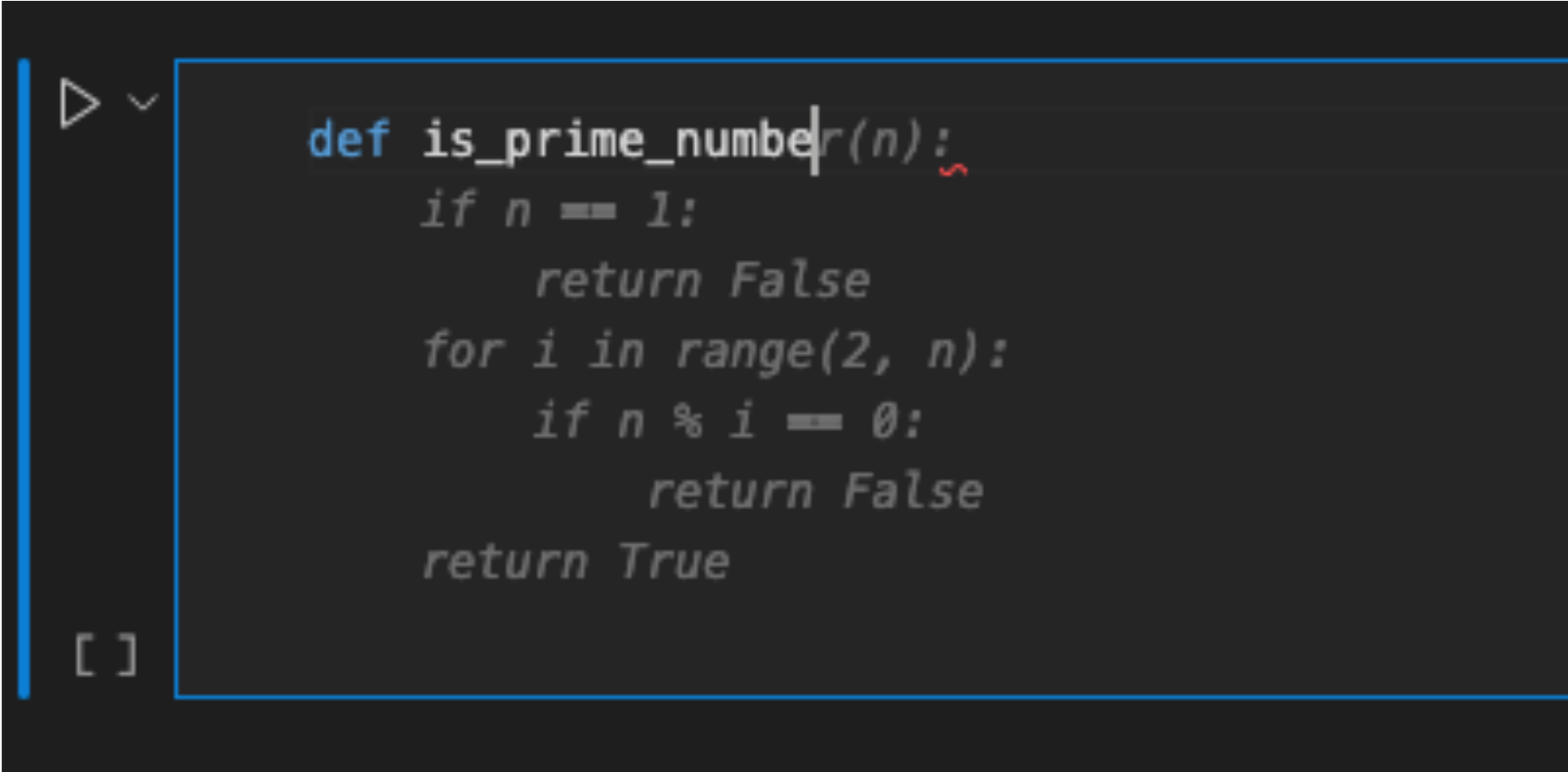
---

- 様々な応用

- 検索：Webページのリスト→質問に対する回答
- コーディング：AIと一緒にコード生成
- 文書生成：メールやワード，エクセルの文面案作成
- 文書閲覧：Webページ，PDF等の要約，補足説明  
⇒ Microsoftは様々なサービスに生成AI技術を搭載中
- 画像・動画生成：自分のニーズに合ったものを生成
- 文書を読んで特定項目を評価：評価基準を明確にしておくとも自動的に評価が可能になる



# VS code + GitHub Copilot (OpenAI) によるコーディング



The image shows a screenshot of the Visual Studio Code editor interface. On the left, there is a sidebar with a file explorer icon (a triangle) and a search icon (a magnifying glass). Below the sidebar, there is a status bar showing the text "[ ]". The main editor area displays a Python function definition for `is_prime_number`. The code is as follows:

```
def is_prime_number(n):  
    if n == 1:  
        return False  
    for i in range(2, n):  
        if n % i == 0:  
            return False  
    return True
```



契約書の内容変更を要望するメールを送りたいのですが、文面を考えてください。



下記のような文面が考えられます。

件名：契約書の内容変更について

拝啓

[相手先名称] 様

この度は契約書に関するご連絡をいただきありがとうございます。

私共 [自分の会社名称] は、当該契約書に基づき、お互いに信頼関係を築くことを目的としております。

ところが、契約書の内容に関して、私共が望む変更点があります。具体的には、[変更点を詳細に説明] です。この変更により、より良い契約関係を築くことができると考えられます。

そこで、この変更を受け入れていただけるかどうかお尋ねさせていただきたく思います。私共からのこの要望に対して、ご賛同いただける場合は、変更内容を記載した修正案をお送りすることができます。

何卒、ご検討の程、よろしくお願い申し上げます。

敬具

[自分の会社名称]

[自分のサインと名前]

## メール文面案作成

# ChatGPTのインパクト

---

- 危険性

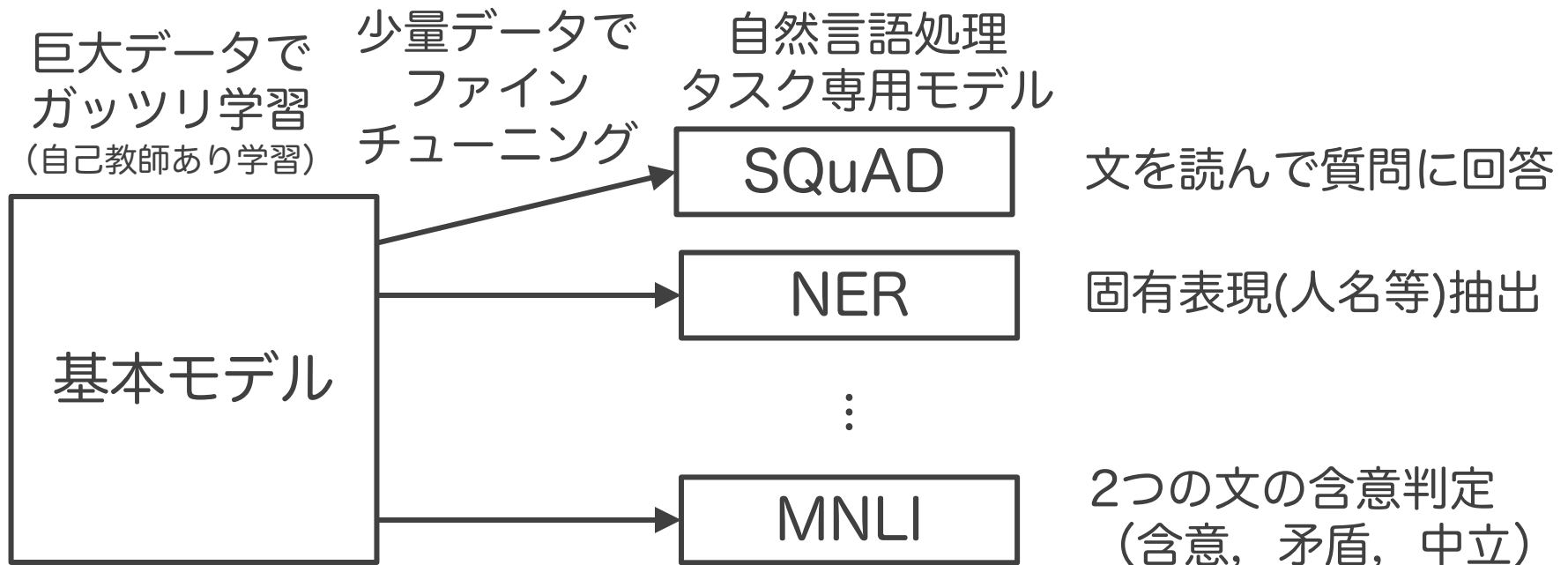
- ChatGPTが生成する誤情報/極論の流布
  - ChatGPTに反論することは難しい
  - GDPR：AIの自動決定に異議を唱える権利
- ChatGPTの間違った指示を鵜呑みにする危険性
  - 機器の誤操作により人が亡くなる等
- アライメント（人間の好みへのモデルの調整）に対する認識不足
  - 研究者やワーカーさんの好みが反映されている

# 大規模言語モデル の基礎技術

---

# 大規模言語モデルの基本モデル

- ひとつの巨大モデルをファインチューニングすることで、様々な自然言語処理タスクを行う



# 大規模言語モデルの基本モデル

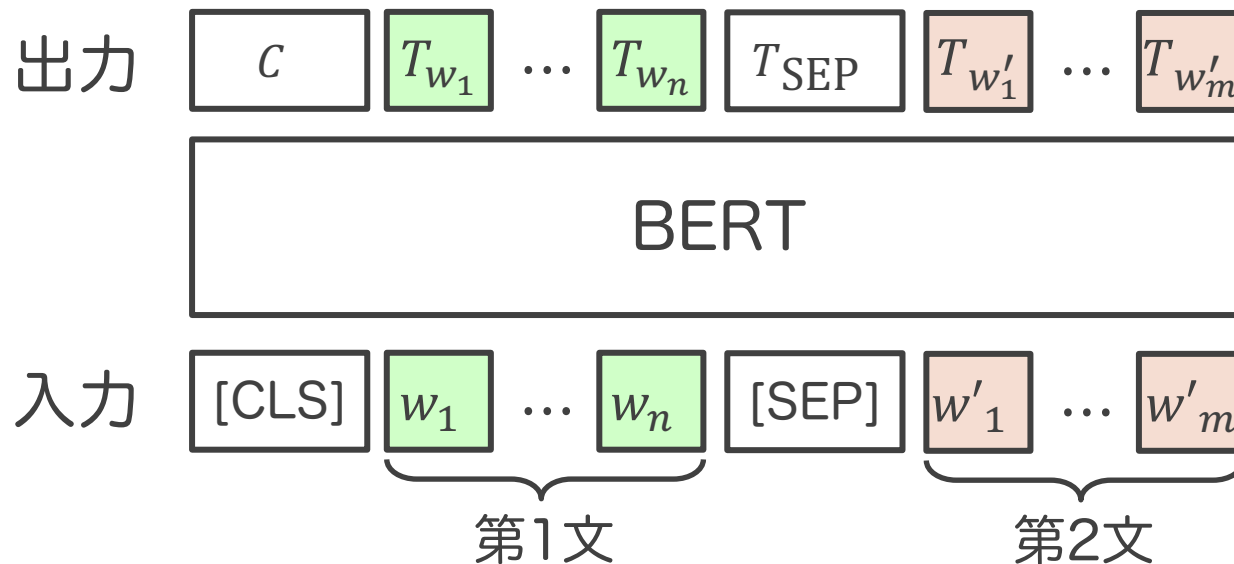
---

## ● 長所

- 基本モデルの学習にはラベルが不要.  
大量の自然言語データがあればよい.  
⇒ 学習済みモデルはHugging Face等で入手可能
  - 少量のラベル付きデータで様々なタスクに対応可能
- 
- 基本モデルは言語自体を学習 (Pre-training)
  - その後のファインチューニングで  
タスクへの対応方法を学習

# 基本モデルの構造（BERTの例）

- Transformerのエンコーダー部分と同様
- 様々なタスクに対応できるようにするため、入力で2つの文を入力する

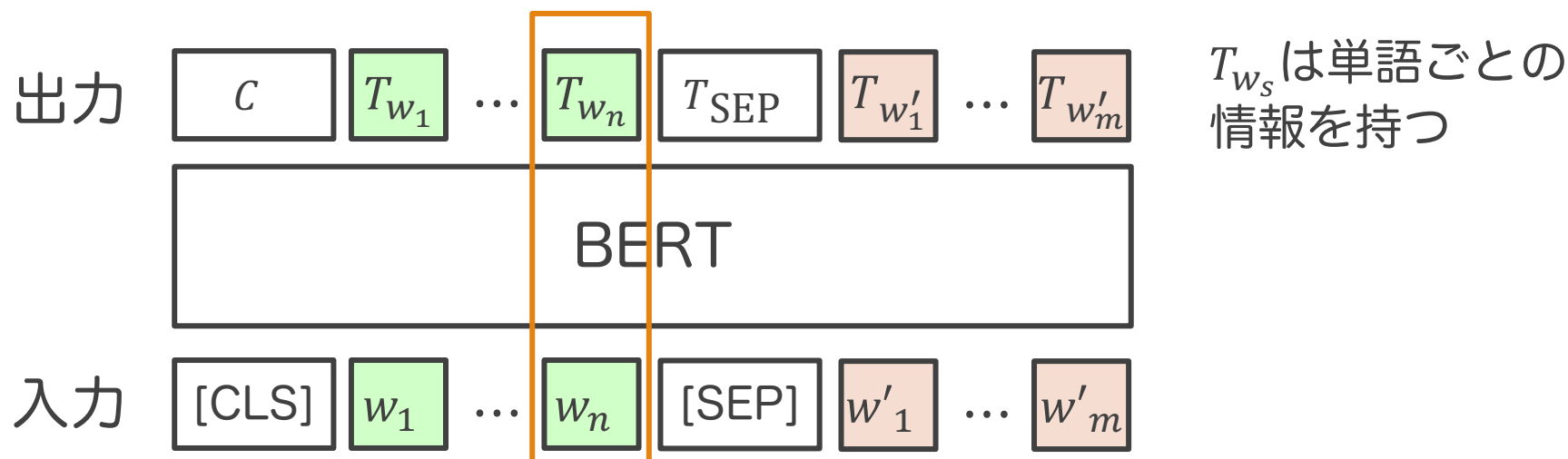


# BERTの学習(Pre-training)

- 2つの問題を使って学習する

1. 穴埋め問題

- 入力単語の15%をマスクし，他の部分から予測する  
⇒  $w_s$  をマスクした時は  $T_{w_s}$  を使って予測を行う



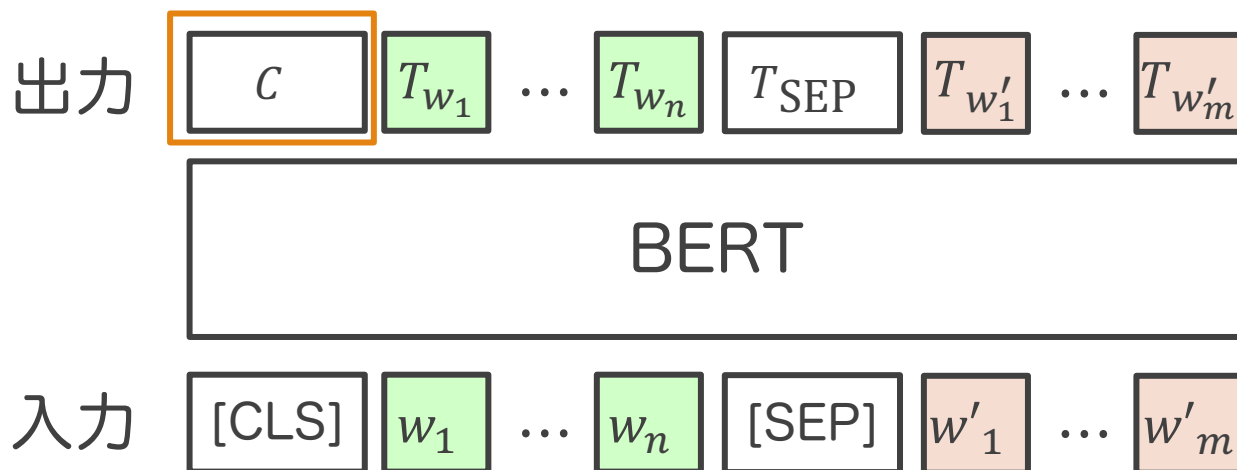


# BERTの学習(Pre-training)

## 2. 次の文予測

- 2つの文を入力し、それが連続する2文からランダムにつなげた2文かを予測する.

⇒  $C$ を使って予測を行う ⇒  $C$ は文全体の情報を持つ

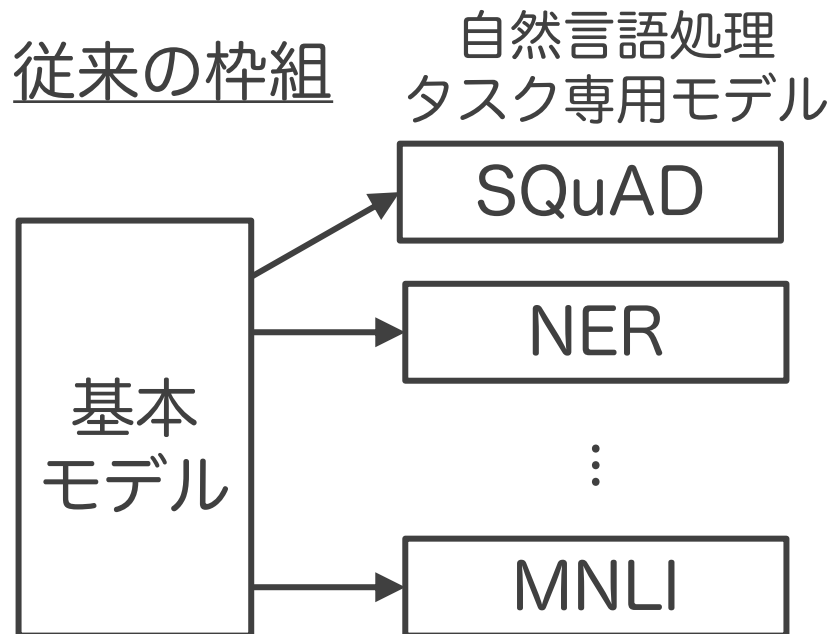


# Text-to-Text

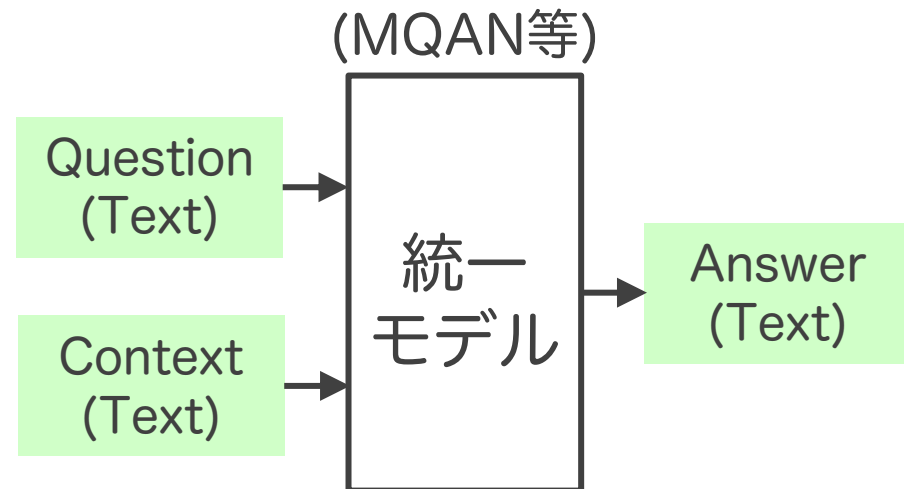
---

# decaNLP (Natural Language Decathlon)

- ひとつのモデルで様々なタスクに対応する枠組
- モデルの入出力をテキストに統一 (text-to-text)



## decaNLP



# decaNLP (Natural Language Decathlon)

Questionでタスクを指定する

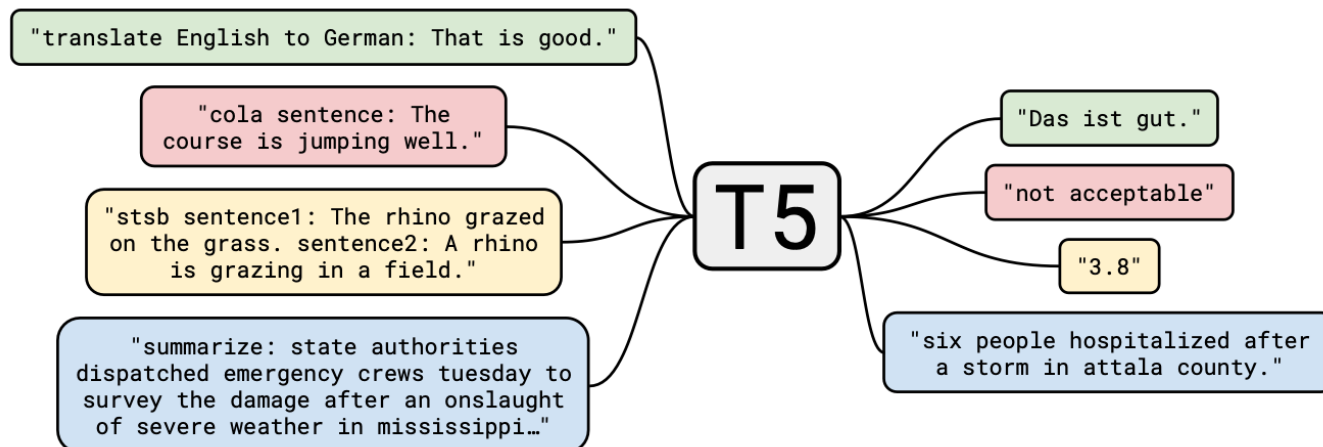
→ 聞き方（文言）次第で性能が変化する → プロンプトチューニングの時代へ  
質問文の調整

## Examples

<u>Question</u>	<u>Context</u>	<u>Answer</u>
What is a major importance of Southern California in relation to California and the US?	...Southern California is a <b>major economic center</b> for the state of California and the US....	<b>major economic center</b>
What is the translation from English to German?	Most of the planet is ocean water.	<b>Der Großteil der Erde ist Meerwasser</b>
What is the summary?	<b>Harry Potter star Daniel Radcliffe</b> gains access to a reported <b>£320 million fortune</b> ...	<b>Harry Potter star Daniel Radcliffe gets £320M fortune...</b>
Hypothesis: Product and geography are what make cream skimming work. <b>Entailment</b> , neutral, or contradiction?	Premise: Conceptually cream skimming has two basic dimensions – product and geography.	<b>Entailment</b>
Is this sentence <b>positive</b> or negative?	A stirring, funny and finally transporting re-imagining of Beauty and the Beast and 1930s horror film.	<b>positive</b>

# T5 (Text-to-Text Transfer Transformer)

- Text-to-Text形式での高性能モデル
  - ただしタスクごとのファインチューニングはあり
- Transformerと同等のエンコーダ-デコーダモデルをベースとする



# アライメント

---

# InstructGPT

---

- GPT3等の既存モデルの問題点
  - 事実を捏造する
  - 偏見や毒のある文を生成する
  - ユーザの指示に従わない
- InstructGPT
  - GPT3を人間の好みに合わせて調整（アライメント）する
    - 教師あり学習やRLHFと呼ばれる強化学習を利用する
    - 評価結果では「事実の捏造」に関して特に大きく改善した

# InstructGPT : アライメントについての論点

---

- 誰の好みの合わせてアライメントするのか？
  - InstructGPTでは雇われたワーカーが「好み」に関するデータを作成する
    - ワーカーは概ね米国と東南アジアの英語話者 ⇒ それで良いのか？
    - ワーカー間での意見が割れている ⇒ 一致したのは73%
  - ワーカーは研究者により選抜され，研究者により指示を受けて作業を行う ⇒ 研究者の好みになってる？
    - 「センシティブな内容かどうか」について研究者と意見が一致するワーカーを採用  
(観点：毒のある発言，性的，暴力的，批判的，政治的)



# Hugging Face Transformers

---

# Hugging Face Transformers

---

- 最新の自然言語処理モデルを活用するためのプラットフォーム
  - 様々な学習済みモデルをロードできる
  - ファインチューニングも行える（今日は省略）
  - モデルに依存しない統一的なインターフェース
  - PyTorch, TensorFlow, JAXに対応

# 使い方

---

- pipeline()

- 最も簡単な使い方
- タスクを指定するだけ
- 内部モデルを自分で指定することもできる
- タスク例
  - "text-generation" : 指定した単語列の後ろに続く文を自動生成する
  - "sentiment-analysis" : 入力した文がポジティブな内容かネガティブな内容かを分類する
  - "question-answering" : 文を読み, それについての質問に答える

# 使い方

---

- TensorFlowモデル生成
  - タスクに合わせてモデルを生成する
  - 学習済みのモデルをロードできる
  - トークナイザー（単語分割器）も同時に指定する必要がある
  - モデルと同じものを使う必要がある