考古遺物型式評価複合AIのための自然言語処理に関する研究
自然言語処理レヴューとMT5モデルの応用

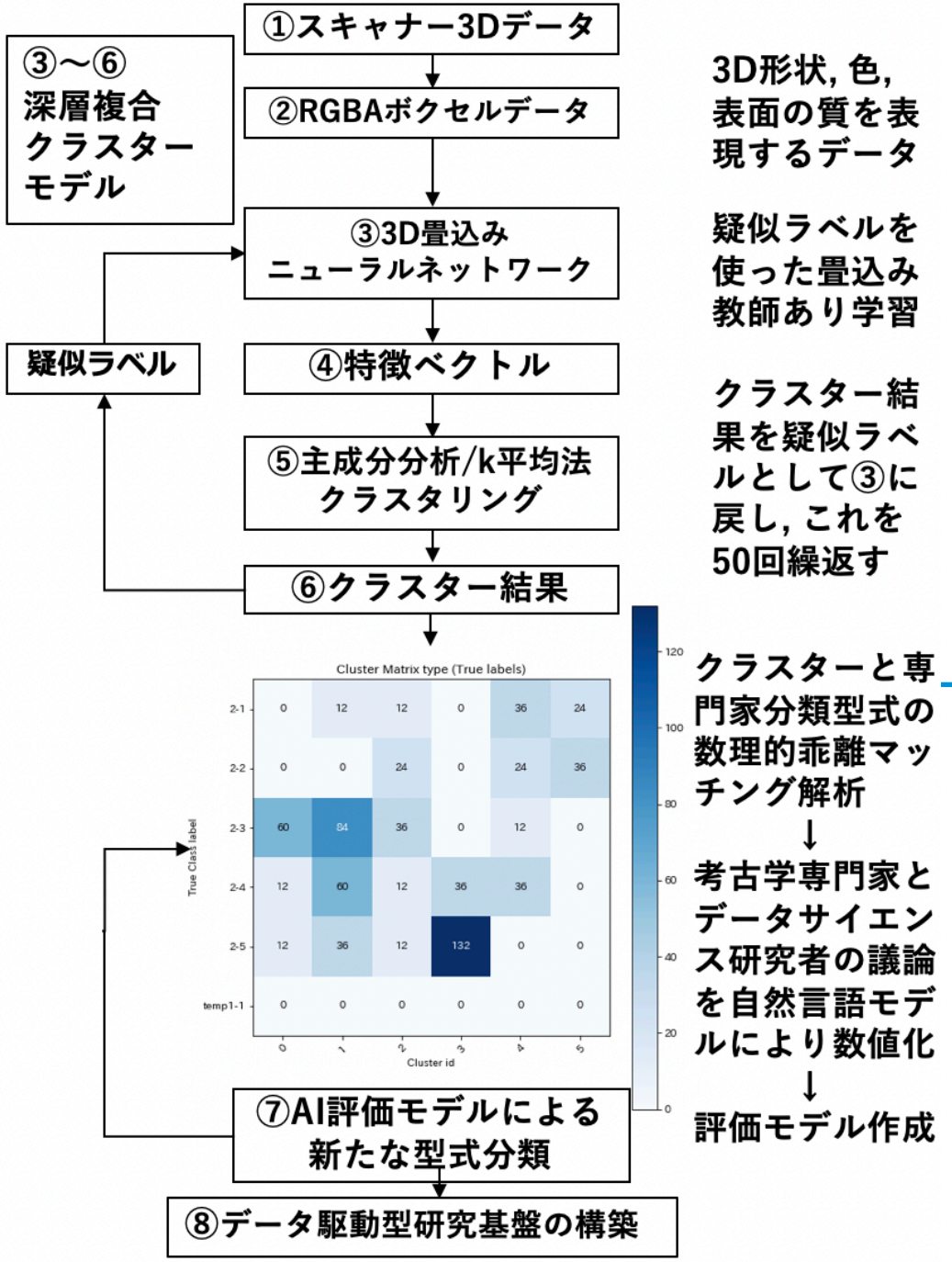
BSNアイネット　　市川健太, 南雲彩花
新潟国際情報大学　　藤田晴啓

2022/6/9 新潟国際情報大学にて発表

研究の目的

- 深層複合クラスターモデルによる出力クラスター
- 考古専門家により分類された報告書記載型式
- 以上の混同行列から数理解析行うアルゴリズムに
- データサイエンス研究者および考古専門家の議論を
- 自然言語処理MT5モデルにて要約，数値化し
- あらたな型式分類を作成するモデルの構築

今回の研究は⑦の処理フロー



第1図　複合AIによるデータ駆動型研究基盤の構築

言語は **系列データ** の性質も持っている

自然言語処理における **構文理解** と **文脈理解** の重要性

✓ 文や語の **順番と依存関係**（構文理解）が重要となる顕著な例：

［道案内］

新潟駅の北口から出て直進し，突きあたりを右折し，最初の信号を左折して直進し，4つ目の交差点を左折すると新潟中央郵便局に到着します.

✓ 文や語の **意味的関連**（文脈理解）が重要となる顕著な例：

［多義語］

- 「かける」：乗算/取り付け/音を出す/粉末や液体を散らす/電話の発信/掃除機/着席/託す…
- 「結構です」：肯定/否定

自然言語処理分野における
深層学習の歴史

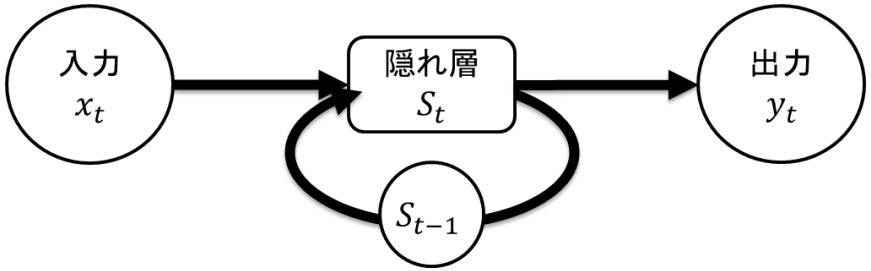
：2017年以前

自然言語処理における深層学習の歴史（1）

2017年以前（1）：RNN（再帰型ニューラルネットワーク）

RNN：Recurrent Neural Network

- NN単体では各入力データは独立で順番や依存関係を処理しない
- RNNでは**過去の入力**を利用することで**順番**と**依存関係**を処理する
- 長い系列データに対しては有効性が薄い
  - 「全て等価に記憶する（丸暗記）」ため過去データの増加につれ1データの重みが縮小
  - 勾配消失/勾配爆発を起こしやすい
- 逐次実行のため計算時間がかかる（**並列処理不可**）

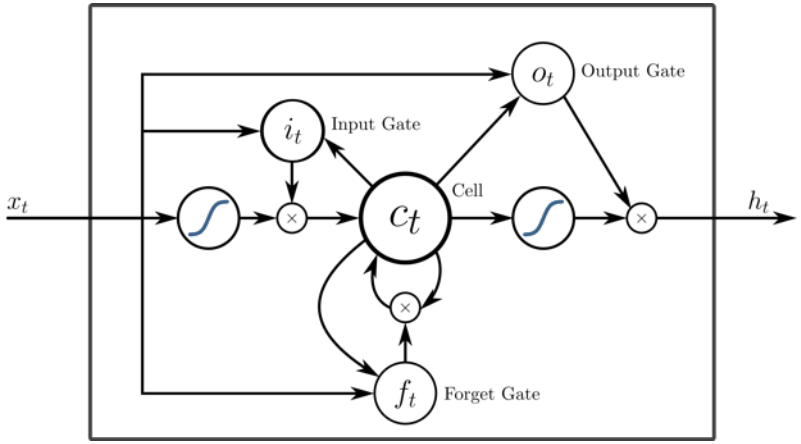


自然言語処理における深層学習の歴史（1.5）

2017年以前（1.5）：LSTM（長・短期記憶）

LSTM：Long Short-Term Memory

- RNNに**忘却ゲート**などを追加
  - 短期記憶を取捨選択し，重要な記憶のみを長期保持する
  - RNNの勾配消失/勾配爆発問題を大幅に軽減
  - （RNNよりは）長い系列データにも有効性
- 長文になると長期の依存関係を取り込めない
  - 更に、長文へ対応させるにつれ構造が複雑になりやすい
  - 複雑になるにつれ解釈可能性も落ちる
- 逐次実行のため計算時間がかかる（**並列処理不可**）



https://en.wikipedia.org/wiki/Long_short-term_memory#/media/File:Peephole_Long_Short-Term_Memory.svg

Long Short-Term Memory; Sepp Hochreiter, Jürgen Schmidhuber; Neural Comput 1997; 9 (8): 1735–1780. doi: https://doi.org/10.1162/neco.1997.9.8.1735

自然言語処理における深層学習の歴史（2）

2017年以前（2）：Attention Mechanism（注意機構）

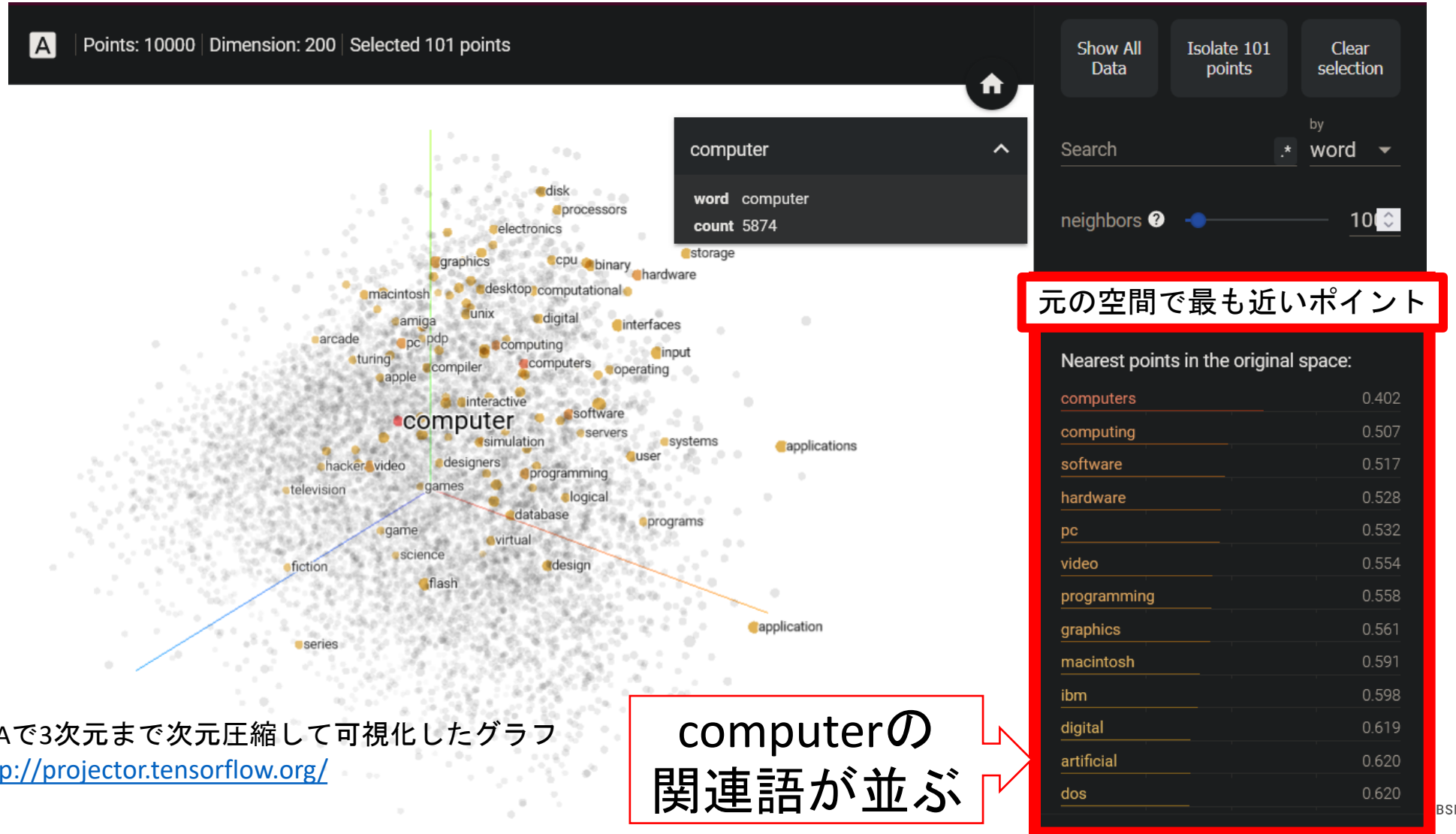
Attention Mechanism

- **入力文章の各単語の分散表現**から重みを計算し、それに基づき**過去データから記憶した各単語の分散表現**から**任意の分散表現**を出力

Neural machine translation by jointly learning to align and translate. / Bahdanau, Dzmitry; Cho, Kyung Hyun; Bengio, Yoshua.2015. Paper presented at 3rd International Conference on Learning Representations, ICLR 2015, San Diego, United States.
arxiv : https://arxiv.org/abs/1409.0473

単語の分散表現

単語の分散表現：単語を数百次元のベクトルで表現



PCAで3次元まで次元圧縮して可視化したグラフ
http://projector.tensorflow.org/

computerの関連語が並ぶ

自然言語処理における深層学習の歴史（2）

2017年以前（2）：Attention Mechanism (注意機構)

Attention Mechanism

- 入力文章の各単語の分散表現から重みを計算し、それに基づき過去データから記憶した各単語の分散表現から任意の分散表現を出力
  - ✓ 入力文章中の各単語と記憶から索引された単語の**関連度**を算出
  - ✓ **入力文章中のどの単語に注意するべきか点数付け**

Neural machine translation by jointly learning to align and translate. / Bahdanau, Dzmitry; Cho, Kyung Hyun; Bengio, Yoshua.2015. Paper presented at 3rd International Conference on Learning Representations, ICLR 2015, San Diego, United States.
arxiv : https://arxiv.org/abs/1409.0473

Attention例

Attentionスコアのヒートマップ：低評価・高評価レビュー

略

- if I can give this restaurant a 0 I will we be just ask our waitress leave because someone with a reservation be wait for our table my father and father-in-law be still finish up their coffee and we have not yet finish our dessert I have never be so humiliated do not go to this restaurant their food be mediocre at best if you want excellent Italian in a small intimate restaurant go to dish on the South Side I will not be go back

---

- i be so angry about my horrible experience at Medusa today my previous visit be amaze 5/5 however my go to out of town and I land an appointment with Stephanie I go in with a picture of roughly what I want and come out look absolutely nothing like it my hair be a horrible ashy blonde not anywhere close to the platinum blonde I request she will not do any of the pop of colour I want and even after specifically tell her I do not like blunt cut my hair have lot of straight edge she do not listen to a single thing I want and when I tell her I be unhappy with the colour she basically tell me I be wrong and I have do it this way no no I do not if I can go from Little Mermaid red to golden blonde in 1 sitting that leave my hair fine I shall be able go from golden blonde to a shade of platinum blonde in 1 sitting thanks for ruin my New Year's with 1 the bad hair job I have ever have

(a) 1 star reviews （低評価）

略

- great food and good service .... what else can you ask for everything that I have ever try here have be great

- first off I hardly remember waiter name because its rare you have an unforgettable experience the day I go I be celebrate my birthday and let me say I leave feel extra special our waiter be the best ever Carlos and the staff as well I be with a party of 4 and we order the potato salad shrimp cocktail lobster amongst other thing and boy be the food great the lobster be the good lobster I have ever eat if you eat a dessert I will recommend the cheese cake that be also the good I have ever have it be expensive but so worth every penny I will definitely be back there go again for the second time in a week and it be even good ..... this place be amazing

(b) 5 star reviews （高評価）

Figure 2: Heatmap of Yelp reviews with the two extreme score.

Yelp reviews；旅行レビューのサイト

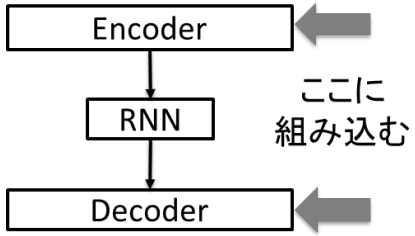
https://arxiv.org/pdf/1703.03130.pdfより引用

自然言語処理における深層学習の歴史（2）

2017年以前（2）：Attention Mechanism (注意機構)

Attention Mechanism

- 入力文章の各単語の分散表現から重みを計算し，それに基づき過去データから記憶した各単語の分散表現から任意の分散表現を出力
  - ✓ 入力文章中の各単語と記憶から索引された単語の**関連度**を算出
  - ✓ **入力文章中のどの単語に注意するべきか**点数付け
- RNNが記憶しきれない過去の情報を保持



注）Attentionの計算手法は複数あるため省略

長文でも単語の**順番**と
依存関係（構文）を掴めるように！

逐次実行問題は未解決
（RNNを使っているため）

Neural machine translation by jointly learning to align and translate. / Bahdanau, Dzmitry; Cho, Kyung Hyun; Bengio, Yoshua.2015. Paper presented at 3rd International Conference on Learning Representations, ICLR 2015, San Diego, United States.
arxiv : https://arxiv.org/abs/1409.0473

自然言語処理における深層学習の歴史（3）

2017年以前（3）：CNNを用いた逐次計算の削減

CNN：Convolutional neural network

- 畳み込み演算で1つの入力データ内での「位置」を処理・認識
  - 計算機性能向上により「1つの入力データ」として入力できる文字列が長くなった
  - Attention Mechanismとの併用
- 単語の順番と依存関係（構文）を処理しつつ部分的に並列処理可能に
- **文章が長くなる（Nの増加）に比例して計算量が増加**
  - O(N) もしくは O(logN)
  - 計算量の制約により長文の依存関係は掴めない

計算時間と長文の依存関係を両立できず行き詰まり

2017-2018年
自然言語処理分野の技術革新

【Transformer】【BERT】の登場

自然言語処理分野の技術革新【Transformer】

2017年：Transformer の登場

Attentionのみを用いた最初のモデル

- 再帰や畳み込み演算（RNN・CNN）を使用せず
  - Attentionは単語の依存関係は処理できるが、単語の順番（位置関係）を処理する方法がなかったためにRNN/CNNと組み合わせていた
- 位置EncodeとSelf-Attentionで単語の順番を認識
  - **並列化が比較的容易**であり計算時間も大幅減
  - **計算量は文章の長さに依らない O(1)**

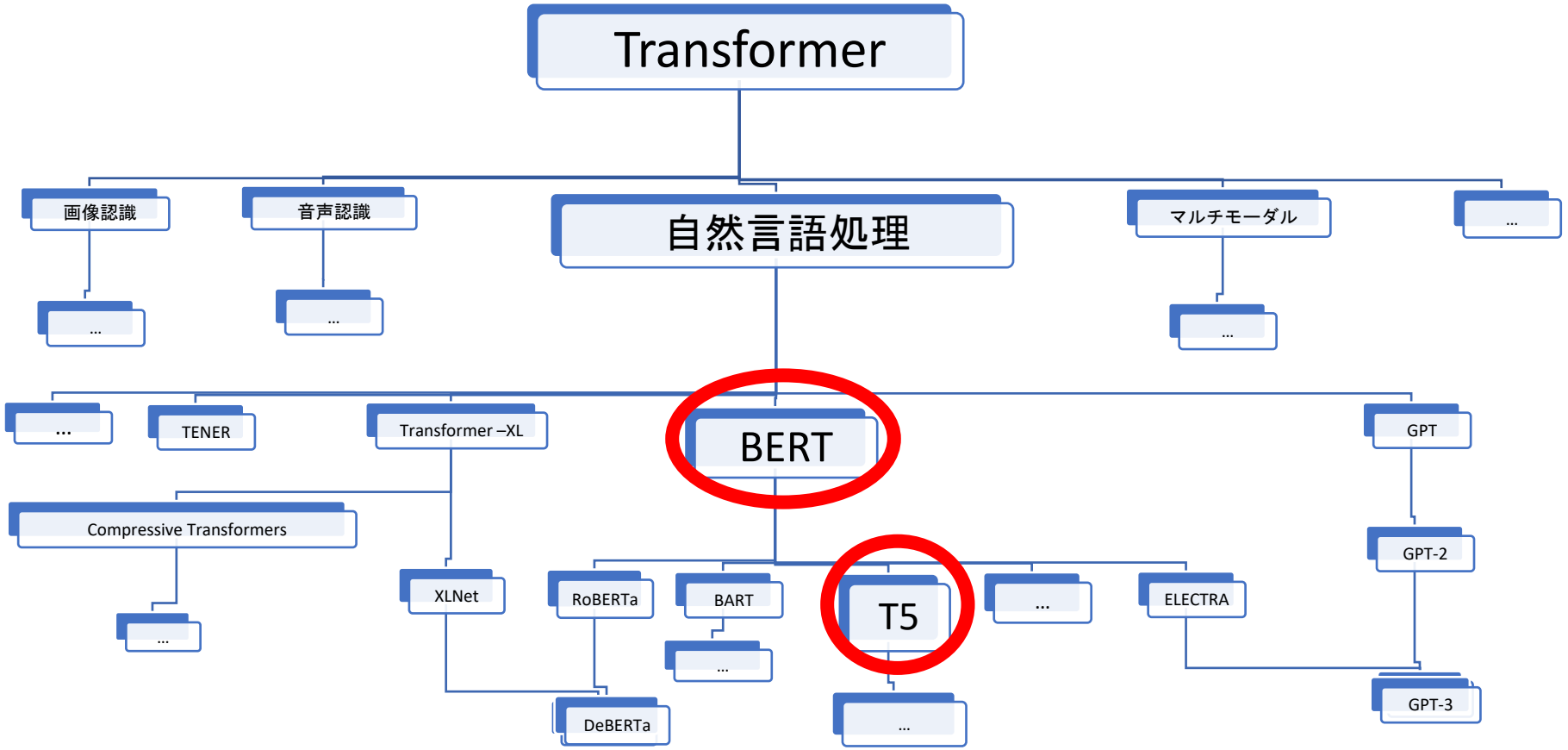
単語の順番と依存関係（構文）を掴みつつ現実的な計算量に！

位置Encode：入力文章の分散表現を入力する際に，各単語位置に一意の値を分散表現へ加算
Self-Attention：入力ベクトル（入力文章の分散表現）から算出した重みと，入力ベクトルと同じベクトルから算出した重みを使用しAttentionを計算することで，単語の順番についての関連性を数値化

Attention is All you Need (Vaswani, Advances in Neural Information Processing Systems) arXiv：https://arxiv.org/abs/1706.03762

自然言語処理分野の技術革新【Transformer】

Transformerの派生（極一部）



注：派生モデルが非常に大量にあるため大幅に省略した図である

補足資料

Transformer のモデル構造図

近年の深層学習モデルは技術的な変遷が非常に速く、また、その多くが複雑な構成をしている。

Transformerのモデル構成も右図の通り複雑である。

本講義では以降で紹介するモデル（BERT / T5 / mT5）も含め、モデル構成までは取り扱わない。

興味がある方はこちらから：

Attention is All you Need (Vaswani, Advances in Neural Information Processing Systems)

arXiv : https://arxiv.org/abs/1706.03762

深層学習界の大前提Transformerの論文解説！

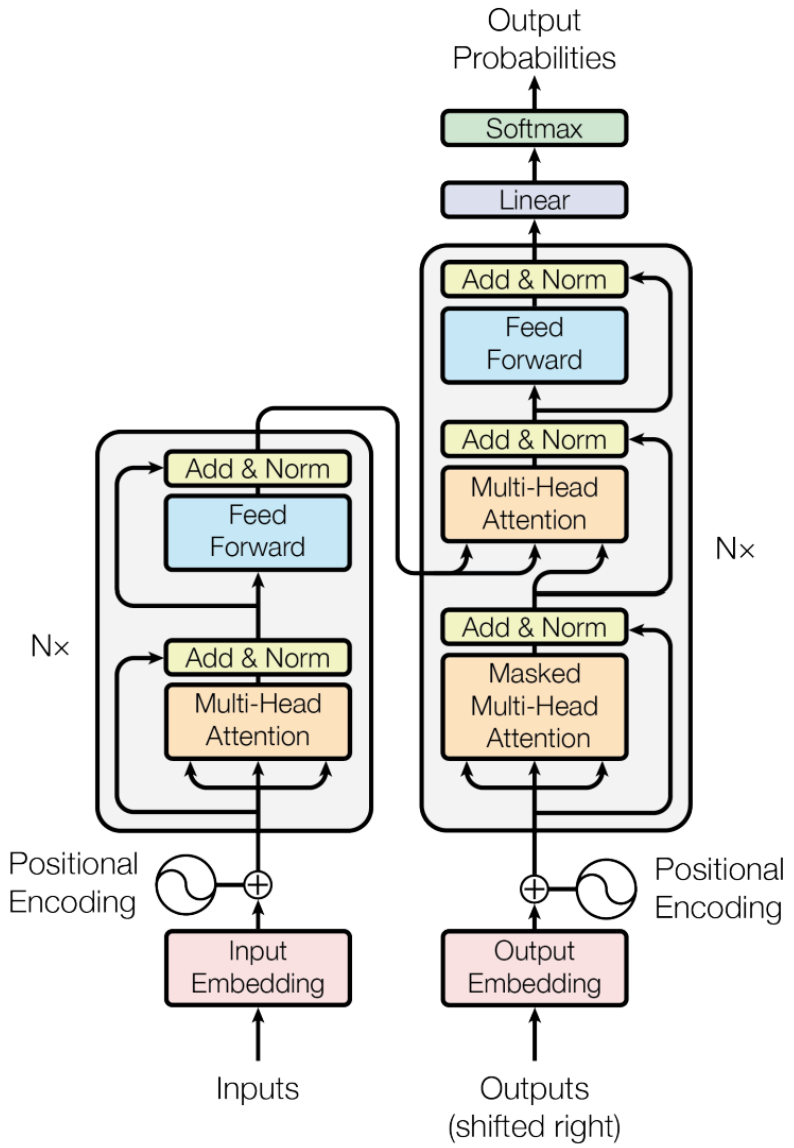


Figure 1: The Transformer - model architecture.

https://arxiv.org/abs/1706.03762 より引用

自然言語処理分野の技術革新【BERT】

2018年：**BERT**（Transformerによる双方向からの符号器表現）

BERT：Bidirectional Encoder Representations from Transformers

- Googleから発表された転移学習を利用する自然言語処理モデル
- 多様な自然言語処理タスクにおいて当時の最高スコアを出した
  - 自然言語処理タスク：翻訳、文書分類、要約、質問応答など自然言語処理の仕事の分野

✓ 文脈理解

- 文章をBidirectional（双方向），文頭と文末から学習することで実現

✓ 少量のデータセットでも高い成果

- 正解ラベルが付与されていないデータセットで事前学習可能

✓ 汎用性

- **転移学習**により様々なタスクに応用可能

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al., NAACL 2019)
arXiv : https://arxiv.org/abs/1810.04805

補足事項；転移学習

Transfer Learning（転移学習）とは

ある領域のタスクで学習させた知識を別領域のタスクの学習へ転用させる技術

- ➢応用範囲が広く，訓練時間の短縮が見込める手法
- ➢転移元との関連性が低いと「負の転移」（精度の低下）を引き起こす場合もあるため注意

Transfer Learning（転移学習）の手法：

1. Feature Extraction（特徴抽出）
   - 事前学習済モデルに新しく訓練する分類器を追加し、事前学習した特徴マップを再利用する。
   - モデル全体を再訓練はせず、事前学習済モデルの特徴マップを凍結（固定）し、新しく追加した分類器の層を訓練する。
2. Fine-Tuning（ファインチューニング/微調整）
   - 事前学習済モデル（ベースモデル）の上位層（出力層に近い層）の幾つかを解凍し、新たに追加された分類器の層とベースモデルの上位層の両方を合わせて訓練する。ベースモデルの高次の特徴表現を微調整することで，特定のタスクへより関連性を持たせることができる。

免責；**「転移学習」「ファインチューニング」の定義にはばらつきがある。**本講義では TensorFlow Tutorials の記載に準ずる。
https://www.tensorflow.org/tutorials/images/transfer_learning

自然言語処理分野の技術革新【BERT】

2018年：**BERT**（Transformerによる双方向からの符号器表現）

BERT：Bidirectional Encoder Representations from Transformers

- Googleから発表された転移学習を利用する自然言語処理モデル
- 多様な自然言語処理タスクにおいて当時の最高スコアを出した
  - 自然言語処理タスク：翻訳、文書分類、要約、質問応答など自然言語処理の仕事の分野

✓ 文脈理解

- 文章をBidirectional（双方向），文頭と文末から学習することで実現

✓ 少量のデータセットでも高い成果

- 正解ラベルが付与されていないデータセットで事前学習可能

✓ 汎用性

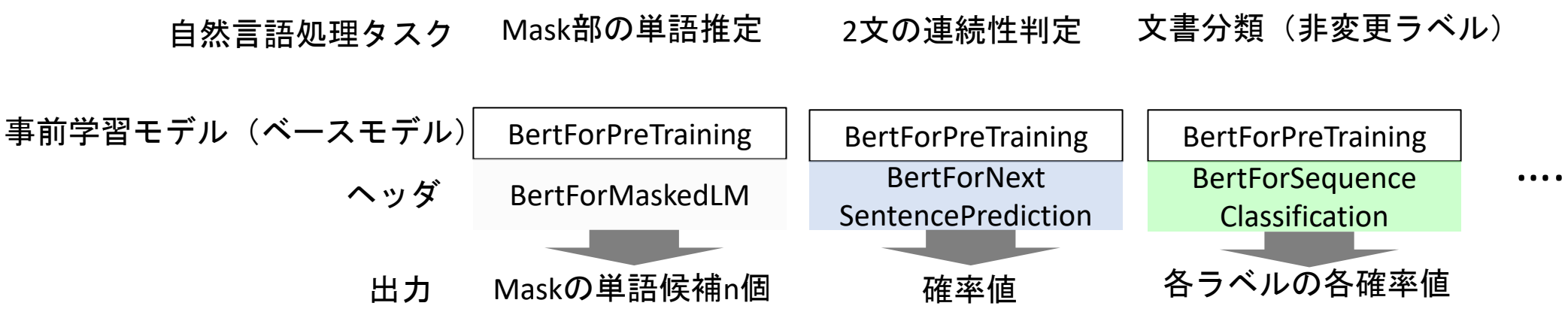
- **転移学習**により様々なタスクに応用可能

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al., NAACL 2019)
arXiv : https://arxiv.org/abs/1810.04805

自然言語処理分野の技術革新【BERT】

BERTの汎用性は利点である一方，

➢ **タスクに応じた header（ヘッダ）を作る必要がある**



いくつかのタスクに応じたヘッダが公開されているが、
目的のタスクと出力が合わない場合は自力でヘッダを作る必要あり

BertLMHeadModel
BertForMaskedLM
BertForNextSentencePrediction

BertForSequenceClassification
BertForMultipleChoice
BertForTokenClassification
BertForQuestionAnswering

補足；ここでの header を訳すならば「モデルの先端」「モデルの頭部」のようになる。なおニューラルネットワークでは入力層に近い層を「下位層」、出力層に近い層を「上位層」と呼ぶ場合がある。

使用するモデル【T5】【mT5】について

2019年：T5（テキストからテキストへ変形する Transformer）

T5：Text-to-Text Transfer Transformer

- Googleから発表された転移学習を利用する自然言語処理モデル
- 分類・翻訳・要約などのタスクを **“Text-to-Text”**へ再構成

✓ **扱いやすいテキスト入力・テキスト出力で統一**

✓ **モデル構成を変えずに全タスクを解く**

https://ai.googleblog.com/2020/02/exploring-transfer-learning-with-t5.html より引用

入力文のタグ付けで
各タスクへ対応

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer (Colin Raffel et al., JMLR, 21(140):1−67, 2020.)
arXiv：https://arxiv.org/abs/1910.10683

Multilingual-T5（mT5）：T5の多言語版モデル

✓ 当初の**T5は英語のみ**対応していた

- 事前学習用データセット（C4）がほぼ英語のデータのみ

✓ **mT5は日本語を含む101言語**に対応

- 101言語で事前学習されたモデルが公開
- 事前学習用データセット（mC4）が101言語対応

26TBのテキストデータ

mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer (Xue et al., NAACL 2021)
arXiv：https://arxiv.org/abs/2010.11934

ハンズオン

XL-Sumデータセットを使用したmT5の転移学習

Hugging Face の紹介

Hugging Face：本講義のデータセットと事前学習済モデルの取得元

米国Hugging Face社が提供する自然言語処理に特化したオープンソースコミュニティ

- ➢オープンソース；ソースコードがGitHub上で公開
- ➢Transformerからの派生モデルを軸として、自然言語処理のタスクを殆どを網羅

提供されているもの：

- ➢深層学習モデルのアーキテクチャ、事前学習済モデル
- ➢自然言語処理関連のデータセット
- ➢…

Hugging Faceが提供するTransformersライブラリにより事前学習済モデルを簡単に使える

https://huggingface.co/
[GitHub] https://github.com/huggingface

事前学習済mT5モデルの読み込み

Hugging Faceから事前学習済mT5モデル：google/mt5-small を読み込む

googleが Hugging Face で公開している下記 mT5モデル はメモリサイズが異なる。

- google/mt5-small（1.12GB）
- google/mt5-base （2.17GB）
- google/mt5-large（4.58GB）
- google/mt5-xl （13.9GB）
- google/mt5-xxl （48.1GB）

今回は一番軽量な【google/mt5-small】を使用する.

精度は他より落ちるが、無料版の Google Colaboratoryで使用できるGPUのメモリサイズとランタイム制約下では、google/mt5-small が扱いやすいため。

https://huggingface.co/google/mt5-small
https://github.com/google-research/multilingual-t5

補足事項

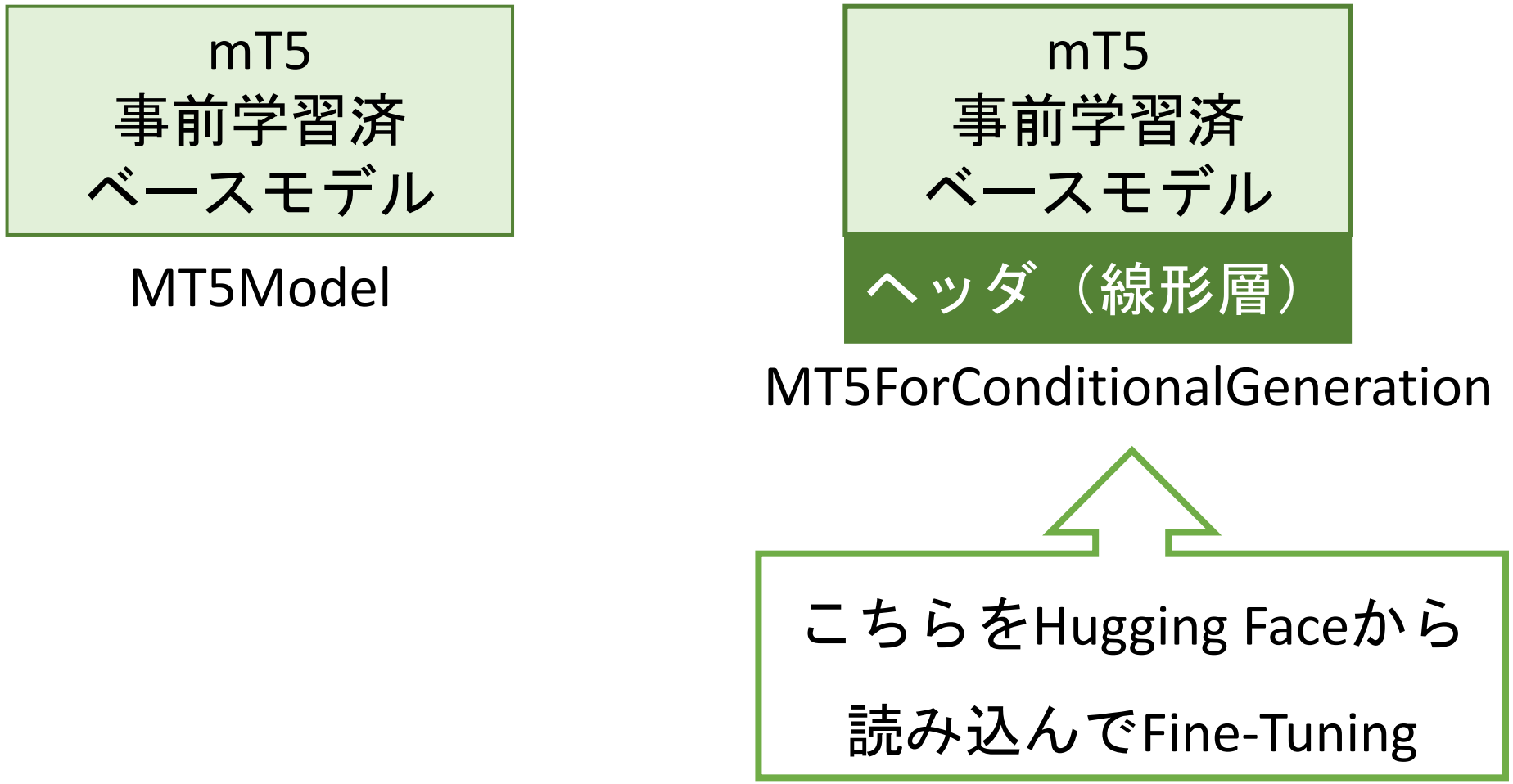
Hugging Faceからのモデル読み込みに関する補足事項

1. MT5Modelクラスと MT5ForConditionalGenerationクラスの違い
   - MT5Modelクラスにはタスク固有のヘッダが含まれておらず、 decoderからはhidden state/隠れ状態が出力として返される。このような生のhidden stateを返すライブラリモデルはベースモデルとも呼ばれる。
   - MT5ForConditionalGenerationクラスには、decoderから最終的なhidden stateを取得する線形層（LM_HEAD）が含まれ、これはトークンを生成する。前述のベースモデルにタスク固有のヘッダが追加されたモデルに相当する。
   - テキスト入力・テキスト出力のタスクへmT5を転移学習させる場合は、MT5ForConditionalGenerationクラスを使用し、別途タスク固有のヘッダを追加したい場合は、MT5Modelクラスを使用する。

2. MT5ForConditionalGenerationクラス、TFMT5ForConditionalGenerationクラス、FlaxMT5ForConditionalGenerationクラスの違い
   - 深層学習をPythonで実装するときに使用するライブラリ/フレームワークによって使い分ける。様々な種類があり、2021年時点で利用者が多いのはPyTorchとTensorFlowの2つ。Hugging Faceでサポートされているのは下記3つ。
     - MT5ForConditionalGeneration：PyTorch
     - TFMT5ForConditionalGeneration：TensorFlow
     - FlaxMT5ForConditionalGeneration：Flax

転移学習

MT5ForConditionalGenerationを読み込んでFine-Tuningを行う



使用データセット

XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages

XL-Sum：44言語の大規模多言語抽象要約

ACL-IJCNLP 2021にて掲載・公開されたBBC News 記事要約データセット

- 本講義では**日本語データセット**のみ使用
- 軽量化と時間短縮のため**先頭15%**のデータのみ使用

本来はニュース記事本文（text）から要約文（summary）を生成する抽象要約タスクを意図されたデータセットですが,

1. 無料版の Google Colaboratory で使用可能なGPUメモリ
2. モデルの訓練にかかる時間

を考慮し、本研究では要約文（summary）から題目（title）を生成します。

XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages (Hasan et al., Findings 2021)
arXiv：https://arxiv.org/abs/2106.13822

データセット（Hugging Face）：
https://huggingface.co/datasets/csebuetnlp/xlsum
データセット（Git Hub）：
https://github.com/csebuetnlp/xl-sum

補足；抽出要約・抽象要約

文書の要約タスクにおいては，大別して2つの手法がある.

1. 抽出型要約

入力文章中から重要な部分をそのまま抽出（抜粋）することで要約文とする手法

- ➢抜粋するため完全に破綻した文章は生成されにくい
- ➢前後の接続が不自然となってしまうことが多い

[注意]

- 要約文として抽出されなかった部分の情報は完全に脱落する
- 訓練データによっては位置バイアスが生じやすい

2. 抽象型要約

入力文章から全体の意味を踏まえ，新たに生成した文を要約文とする手法

近年Encorder-Decorderモデル等のニューラル言語モデルで扱われることが多い

入力文章中には存在しない情報（文字）でも事前学習や訓練データ中の情報は要約文へ反映されうる

- ➢上手くいけば人間から見ても自然な要約文を生成できる
- ➢文章として完全に破綻したものを生成してしまう事態が起こりうる

[注意]

- 不要な情報/不適切な情報を多く生成するよう訓練されてしまう場合がある
- 同じ情報を何度も繰り返すよう訓練されてしまう（モード崩壊）場合がある

使用データセット

XL-Sum: Large-Scale Multilingual Abstractive Summarization for 44 Languages のデータセット構成

id	'48590588'
summary	'イギリスでは10日、与党・保守党の党首選の立候補が締め切られ、10人の議員が正式に出馬を表明した。'
text	'text': '英保守党党首選の立候補者が確定した。 【中略】 スチュアート国際開発相はソーシャルメディアを使ったキャンペーンで、保守党外から注目と称賛を集めている。 しかしこの党首選の行方は、あくまで保守党議員と保守党員が決めるものだ。 （英語記事 Final 10 named for Tory leadership race）'
title	'英保守党党首選、立候補者は10人 第1回投票は13日'
url	'https://www.bbc.com/japanese/48590588'

補足；

前スライドのtext全文：Google ColabのGPUメモリサイズで扱うには長過ぎる.

'text': '英保守党党首選の立候補者が確定した。上段左からマイケル・ゴーヴ環境相、マット・ハンコック保健相、マーク・ハーパー前下院院内幹事長、ジェレミー・ハント外相、サジド・ジャヴィド内相。下段左からボリス・ジョンソン前外相、アンドレア・レッドソム前下院院内総務、エスター・マクヴェイ前雇用・年金相、ドミニク・ラーブ前EU離脱相、ローリー・スチュワート国際開発相　党首選は、テリーザ・メイ首相が7日に保守党党首を退任したことに伴うもの。保守党は今後、議員による投票で立候補者を2人まで絞った後、全国の保守党員による決選投票を行う。　次期党首は自動的にイギリスの首相となる。メイ首相は後任が決まるまで首相にとどまる。　立候補者は以下の10人。　ハント外相やラーブ前EU離脱相、ゴーヴ環境相は立候補の締め切り前から選挙活動を始めていたが、いずれも正式に出馬が認められている。　一方、サム・ジーマー前閣外相は支持を固めるには時間が足りなかったとして、出馬を取り消した。　ジーマー氏は出馬に意欲を示していた議員の中では唯一、欧州連合（EU）離脱問題で2度目の投票を支持していた。　＜関連記事＞　党首選では通常、2人の推薦人がいれば出馬できるが、保守党は6月初め、手続きを加速させるために出馬には6人の支持者が必要というルールを追加した。　さらに、13日に行われる最初の議員投票では最低16人、18日の2度目の投票では最低32人の支持者を得なければ次の投票に進めないというルールも追加された。　議員投票は13日、18日、19日、20日に行われ、候補者は最終的に2人に絞られる。　その後、22日からイギリス全土の保守党党員約16万人による郵便投票が行われ、4週間後の7月22日の週にイギリスの新首相が発表される予定だ。　ゴーヴ環境相はジョンソン氏を批判　ゴーヴ環境相は演説で、「国家サイバークライム対策チーム」の設置や法的手段による抗議から軍を守る施策など、首相として進めていきたい政策を次々と明らかにした。　また、国民保健サービス（NHS）への予算充当を確約したほか、ブレグジット（イギリスのEU離脱）については「実現できると思えるほどブレグジットへの信頼が集まっていなかった」と述べた上で、自分は「真っ当な計画」を持っていると示唆した。　一方、ジョンソン前外相が年収5万ポンド（約700万円）以上の人の所得税を減らす公約を掲げたことについて、「税制や福祉のシステムを使ってすでに裕福な人に減税措置を施すことなど、私が首相になったら絶対にやらない」と批判した。　ジョンソン氏はまだ自身の選挙活動について、TVインタビューを受けていない。　ハンコック氏とハント氏には強力な後押し　ハンコック保健相は、メイ首相の右腕とされたデイヴィッド・リディングトン内閣府担当相の支持を取り付けた。リディングトン氏はBBCのローラ・クンスバーグ政治編集長に、ハンコック氏は明確な未来のビジョンを持っていると話した。　ハンコック氏は、全国生活賃金を1時間当たり10ポンド（約1380円）以上にする公約を明らかにしている。　一方、ハント外相にはペニー・モーダント国防相とアンバー・ラッド雇用・年金相が支持を表明している。　ハント氏は、ブレグジットの膠着（こうちゃく）状態を脱するには「とても賢明な」アプローチが必要だと話し、「空論」ではなく「経験豊富で真面目な首相」が必要とされていると述べた。　ハント氏をめぐっては、人工妊娠中絶が可能な時期を24週から12週に短縮すべきだという「個人的な意見」が批判されたが、これについてハント氏は、首相になっても中絶期限短縮の法改正を行うつもりはないとしている。　その他の立候補者の動向は以下の通り。　＜分析＞多くの候補者、異なるアピール　――ベン・ライトBBC政治担当編集委員　保守党議員は次の党首に何を求めているのだろう？　総選挙に勝利し、自分たちの議席を守ってくれる人だ、もちろん。それからブレグジットについて有望な案を持っている人物。むっつり意気消沈した保守党を生き返らせてくれる人物。　議会で輝く人物が求められているなら、ゴーヴ環境相が快勝するだろう。しかしゴーヴ氏は前回の党首選でジョンソン前外相の出馬を阻止したし、最近では20年前のコカイン使用を認めて謝罪しており、その評判は傷ついている。　そのジョンソン氏は保守党内で対立しがちで、私生活もずっとごたごたしている。しかしそれでも、ジョンソン氏はイギリスで最も認知度が高く、カリスマ性あふれる政治家の1人だ。　ハント外相は明確な意見を持ち、管理職らしい言動だ。元弁護士のラーブ前EU離脱相は物事を一刀両断するような熱意を持っているし、ジャヴィド氏は保守党の頂点まで上り詰めた人物だ。　マクヴェイ前雇用・年金相はテレビのキャスターとしてキャリアを積んだ。レッドソム前下院院内総務にとっては2度目の党首選だ。スチュアート国際開発相はソーシャルメディアを使ったキャンペーンで、保守党外から注目と称賛を集めている。　しかしこの党首選の行方は、あくまで保守党議員と保守党員が決めるものだ。　（英語記事 Final 10 named for Tory leadership race）'

使用データセット

BCC News 記事要約文から題目を生成（1行要約）

記事要約文（入力/学習データ）

'summary'：
'イギリスでは10日、与党・保守党の党首選の立候補が締め切られ、10人の議員が正式に出馬を表明した。'

mT5で転移学習

題目（出力/正解データ）

'title':
'英保守党党首選、立候補者は10人 第1回投票は13日'

要約文から題目を予測するよう転移学習させる

※本来は記事本文から要約文を生成することを意図したデータセットであるため、要約文には存在しない情報が題目に含まれているデータもある

https://www.bbc.com/japanese/48590588

使用データセット

BCC News 記事要約文から題目を生成（1行要約）

記事要約文（入力/学習データ）

'summary'：
'イギリスでは10日、与党・保守党の党首選の立候補が締め切られ、10人の議員が正式に出馬を表明した。'

mT5で転移学習

題目（出力/正解データ）

'title':
'英保守党党首選、立候補者は10人 第1回投票は13日'

要約文から題目を予測するよう転移学習させる

※本来は記事本文から要約文を生成することを意図したデータセットであるため、要約文には存在しない情報が題目に含まれているデータもある

https://www.bbc.com/japanese/48590588

モデルの入力と出力

全体像

summarize: ブレグジット(イギリスの欧州連合離脱)をめぐり、イギリスのトニー・ブレア元首相が、2度目の国民投票を求める主張を再び繰り広げている。可能性があるブレグジットの形態1つ1つについて議会で投票し、どれも合意に達しない場合に実施すべきという。

MT5Tokenizer

トークナイズ

B-1. 入力データ/input_ids（Aの文字列がトークナイズされたトークンID列）

```
tensor([196098,  10701,    267,    259, 112024,  13989,  12774,  28119,    312,
        120176,    423, 165653, 173734,  57490,  21886,    271,    574, 233704,
          3518,    292, 120176,    423,   5408,  36643,    626, 112024,   5832,
          4074, 129321,    536,  57815,   5573,  71658,  65953,  87431, 138988,
        123350,    574, 120203, 175914,  34700,  28241,  13682,    306, 191423,
        112024,  13989,  12774,  28119,    423, 133990,    353,   7513,    353,
          7513,   6719, 143677,    627,  87431,   1604,    292, 188840, 198514,
           600,  23596,  29285, 156479,  67372,  68052,   3104,    306,      1,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0])
```

B-2. 入力データ/attention_mask（モデルが注意を払うべきトークンの判別用）

```
tensor([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
        0, 0, 0, 0, 0, 0, 0, 0])
```

MT5ForConditionalGeneration

mT5モデルへ入力

generate (token ids)：

```
tensor([[ 0, 1612, 15201, 38145, 104476, 3556, 836, 259, 112024, 13989, 12774, 28119, 423, 65953, 87431, 48793, 106547, 40439, 66498, 1]])
```

Decode

（ここではトークナイズされたトークンID列をテキスト表現に戻すこと）

generate：

【英総選挙2019】 ブレグジットの国民投票向け議論始まる

summaryとtitleデータをトークン化（tokenize）する

tokenize：文をtoken/トークンという単位に区切ること

tokenの区切り方には種類がある

- 従来は**単語**や**形態素**で区切られる（前回授業）ことが大半だった
- 近年は単語よりも更に小さい単位で分割した **subword** に類するものが多い

前回授業：単語・形態素による区切り

分かち書き

買ってよかったと感じました。

↓ 分かち書き
単語ごとに分割し、
半角スペースで連結

買っ て よかっ た と 感じ まし た 。

summaryとtitleデータをトークン化（tokenize）する

subword / サブワードの革命的な点

従来の単語/形態素分割：

高頻度語のみに限定し、低頻度語は切り捨てる。

- 計算量の問題で大規模な語彙を扱えなかった
- 切り捨てられた低頻度語は **全て <unk> （unknown）** とされていた

subword：

1. 文を単語分割して各単語の出現頻度を求める
2. 高頻度語は1単語、低頻度語はより短い単位：文字や部分文字列へ分割

低頻度語も最終的には **文字の並び** として落とし込めるように！

summaryとtitleデータをトークン化（tokenize）する

SentencePiece とは

本講義で使用するmT5のtokenizerは、subwordを土台とした「SentencePiece」に基づく

subword： **spaceによる単語区切りがない言語では、言語毎に複雑なルールで分割していた**

- 最初に「単語分割」を行って各単語の出現頻度を求めるため
- 多言語対応が困難
- 言語によっては単語分割の正確性にも疑問符がつく

SentencePiece：**spaceも文字として扱う**

- spaceによる単語分割を行うことなく、言語モデルに基づく教師なし分割を学習する。

SentencePiece：
Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates (Kudo, ACL 2018)
論文著者による紹介（Qiita）：
Sentencepiece：ニューラル言語処理向けトークナイザ

summaryとtitleデータをトークン化（tokenize）する

MT5Tokenizer が行う処理

summarize: ブレグジット(イギリスの欧州連合離脱)をめぐり、イギリスのトニー・ブレア元首相が、2度目の国民投票を求める主張を再び繰り広げている。可能性があるブレグジットの形態1つ1つについて議会で投票し、どれも合意に達しない場合に実施すべきという。

MT5Tokenizer

B-1. 入力データ/input_ids（Aの文字列がトークナイズされたトークンID列）

tensor([196098, 10701, 267, 259, 112024, 13989, 12774, 28119, 312,
120176, 423, 165653, 173734, 57490, 21886, 271, 574, 233704,
3518, 292, 120176, 423, 5408, 36643, 626, 112024, 5832,
4074, 129321, 536, 57815, 5573, 71658, 65953, 87431, 138988,
123350, 574, 120203, 175914, 34700, 28241, 13682, 306, 191423,
112024, 13989, 12774, 28119, 423, 133990, 353, 7513, 353,
7513, 6719, 143677, 627, 87431, 1604, 292, 188840, 198514,
600, 23596, 29285, 156479, 67372, 68052, 3104, 306, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0])

Input ids :
パディング（0埋め）済のトークンID列

B-2. 入力データ/attention_mask（モデルが注意を払うべきトークンの判別用）

tensor([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0])

attention mask :
初期状態では文字の存在する部分が1

summaryとtitleデータをトークン化（tokenize）する

補足；パディング済のトークンID列を文字列へ変換（decode）すると

パディング済のトークンID列

B-1. 入力データ/input_ids（Aの文字列がトークナイズされたトークンID列）

```
tensor([196098,  10701,    267,    259, 112024,  13989,  12774,  28119,    312,
        120176,    423, 165653, 173734,  57490,  21886,    271,    574, 233704,
          3518,    292, 120176,    423,   5408,  36643,    626, 112024,   5832,
          4074, 129321,    536,  57815,   5573,  71658,  65953,  87431, 138988,
        123350,    574, 120203, 175914,  34700,  28241,  13682,    306, 191423,
        112024,  13989,  12774,  28119,    423, 133990,    353,   7513,    353,
          7513,   6719, 143677,    627,  87431,   1604,    292, 188840, 198514,
           600,  23596,  29285, 156479,  67372,  68052,   3104,    306,      1,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0,      0,      0,      0,      0,      0,      0,      0,
             0,      0])
```

decode

パディング済のトークンID列を文字列へ変換

summarize: ブレグジット(イギリスの欧州連合離脱)をめぐり、イギリスのトニー・ブレア元首相が、2度目の国民投票を求める主張を再び繰り広げている。可能性があるブレグジットの形態1つ1つについて議会で投票し、どれも合意に達しない場合に実施すべきという。
</s> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
<pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
<pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
<pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
<pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad> <pad>
<pad> <pad>

</s>：文末位置を示す特殊記号
<pad>：パディングを示す特殊記号

前回の授業時とは異なり、短い場合は末尾がパディング、長い場合は末尾が切り捨てられる。

前回授業題材は短文レビューであり、述語の情報が非常に重要となるため、先頭切り捨てとしていた。
本題材はニュース要約文であり、最初に重要な情報（概要）が存在する可能性が高いため、末尾切り捨てとしている。

事前学習だけのmT5モデルで要約させてみる

転移学習（特定のタスクへモデルを微調整）していない状態で要約させてみる

入力文字列：

summarize:ブレグジット（イギリスの欧州連合離脱）をめぐり、イギリスのトニー・ブレア元首相が、2度目の国民投票を求める主張を再び繰り広げている。可能性があるブレグジットの形態1つ1つについて議会で投票し、どれも合意に達しない場合に実施すべきという。

出力文字列：

<extra_id_0>。)に投票する <extra_id_10>.)) <extra_id_11>.

補足：

mT5の転移学習（特定タスクへのモデル微調整）に関する補足事項

1. T5の例（右図）では文章が出ているのに、前スライドのmT5では出力された文が崩壊している理由
   - Hugging Faceでgoogleから公開されているT5/mT5モデルの相違点は幾つかあるが，内1つに下流タスクで事前学習されているか否かが挙げられる。
   - T5は下流タスクも事前学習されているため、英語の要約タスクの場合、事前学習のみでも読める精度で英語の要約文が生成される。しかし、mT5は下流タスクでの事前学習は行われていないため、転移学習（特定タスクへのモデル微調整）が必須となる。

2. 下流タスクを指定するタグ：prefix （本講義ではsummarize:）について
   - T5はprefixを含めた事前学習が行われているが、mT5はprefixなしで事前学習されている。
   - そのためmT5の場合は、単一タスクへの転移学習でprefixを使用することに利点はない。
   - ただし、複数タスクへの転移学習ではprefixが必要となる。
   - 本講義資料では単一タスクへの転移学習であるが、参考としてprefixを入れている。

T5

mT5モデルを転移学習させる

TransformasのSeq2SeqTrainingArgumentsについて

今回のmT5転移学習でパラメータを設定しているSeq2SeqTrainingArgumentsの引数を一部説明する

- Seq2SeqTrainingArgumentsの引数は90個あるため、ここでは一部のみ紹介
- 他の引数を確認してみたい方は、下記URLを参照

https://huggingface.co/docs/transformers/main_classes/trainer

class transformers.Seq2SeqTrainingArguments <source>

(output_dir: str, overwrite_output_dir: bool = False, do_train: bool = False, do_eval: bool = False, do_predict: bool = False, evaluation_strategy: IntervalStrategy = 'no', prediction_loss_only: bool = False, per_device_train_batch_size: int = 8, per_device_eval_batch_size: int = 8, per_gpu_train_batch_size: typing.Optional[int] = None, per_gpu_eval_batch_size: typing.Optional[int] = None, gradient_accumulation_steps: int = 1, eval_accumulation_steps: typing.Optional[int] = None, eval_delay: typing.Optional[float] = 0, learning_rate: float = 5e-05, weight_decay: float = 0.0, adam_beta1: float = 0.9, adam_beta2: float = 0.999, adam_epsilon: float = 1e-08, max_grad_norm: float = 1.0, num_train_epochs: float = 3.0, max_steps: int = -1, lr_scheduler_type: SchedulerType = 'linear', warmup_ratio: float = 0.0, warmup_steps: int = 0, log_level: typing.Optional[str] = 'passive', log_level_replica: typing.Optional[str] = 'passive', log_on_each_node: bool = True, logging_dir: typing.Optional[str] = None, logging_strategy: IntervalStrategy = 'steps', logging_first_step: bool = False, logging_steps: int = 500, logging_nan_inf_filter: bool = True, save_strategy: IntervalStrategy = 'steps', save_steps: int = 500, save_total_limit: typing.Optional[int] = None, save_on_each_node: bool = False, no_cuda: bool = False, seed: int = 42, data_seed: typing.Optional[int] = None, bf16: bool = False, fp16: bool = False, fp16_opt_level: str = 'O1', half_precision_backend: str = 'auto', bf16_full_eval: bool = False, fp16_full_eval: bool = False, tf32: typing.Optional[bool] = None, local_rank: int = -1, xpu_backend: typing.Optional[str] = None, tpu_num_cores: typing.Optional[int] = None, tpu_metrics_debug: bool = False, debug: str = '', dataloader_drop_last: bool = False, eval_steps: typing.Optional[int] = None, dataloader_num_workers: int = 0, past_index: int = -1, run_name: typing.Optional[str] = None, disable_tqdm: typing.Optional[bool] = None, remove_unused_columns: typing.Optional[bool] = True, label_names: typing.Optional[typing.List[str]] = None, load_best_model_at_end: typing.Optional[bool] = False, metric_for_best_model: typing.Optional[str] = None, greater_is_better: typing.Optional[bool] = None, ignore_data_skip: bool = False, sharded_ddp: str = '', fsdp: str = '', fsdp_min_num_params: int = 0, deepspeed: typing.Optional[str] = None, label_smoothing_factor: float = 0.0, optim: OptimizerNames = 'adamw_hf', adafactor: bool = False, group_by_length: bool = False, length_column_name: typing.Optional[str] = 'length', report_to: typing.Optional[typing.List[str]] = None, ddp_find_unused_parameters: typing.Optional[bool] = None, ddp_bucket_cap_mb: typing.Optional[int] = None, dataloader_pin_memory: bool = True, skip_memory_metrics: bool = True, use_legacy_prediction_loop: bool = False, push_to_hub: bool = False, resume_from_checkpoint: typing.Optional[str] = None, hub_model_id: typing.Optional[str] = None, hub_strategy: HubStrategy = 'every_save', hub_token: typing.Optional[str] = None, hub_private_repo: bool = False, gradient_checkpointing: bool = False, include_inputs_for_metrics: bool = False, fp16_backend: str = 'auto', push_to_hub_model_id: typing.Optional[str] = None, push_to_hub_organization: typing.Optional[str] = None, push_to_hub_token: typing.Optional[str] = None, mp_parameters: str = '', auto_find_batch_size: bool = False, full_determinism: bool = False, sortish_sampler: bool = False, predict_with_generate: bool = False, generation_max_length: typing.Optional[int] = None, generation_num_beams: typing.Optional[int] = None)

mT5モデルを転移学習させる

TransformasのSeq2SeqTrainingArguments（主要）
【基礎編１】

引数名	型	必須 / 選択	初期設定値 /default	説明
output_dir	str	required		モデルの予測とチェックポイントが書き込まれる出力ディレクトリ
per_device_train_batch_size	int	optional	8	訓練用のGPU/TPUコア/CPUあたりのバッチサイズ
per_device_eval_batch_size	int	optional	8	評価/検証用のGPU/TPUコア/CPUあたりのバッチサイズ
num_train_epochs	float	optional	3.0	実行する訓練Epochの総数を指定する。整数値でない場合、最終Epochは小数部分の百分率で実行される。
max_steps	int	optional	-1	実行する訓練stepの総数を正の整数で設定する。上記のnum_train_epochs を**上書き**するため同時に設定しないこと。

mT5モデルを転移学習させる

TransformasのSeq2SeqTrainingArguments（主要）
【基礎編２】

引数名	型	必須 / 選択	初期設定値 /default	説明
evaluation_strategy	str	optional	“no”	訓練中の評価/evaluation（検証/validation）を行う方法 "no" ：訓練中に評価/検証を行わない “steps” ：評価/検証をeval_steps毎に行いlogに記録する “epoch”：評価/検証を各epochの終了後に行う
eval_steps	int	optional	logging_steps	evaluation_strategy=“steps”とした場合は、この引数で訓練中の評価/検証を更新するstep数を指定する。 defaultでは、logging_stepsと同じ値が設定される。
logging_strategy	str	optional	“steps”	訓練中にlogging（記録）を行う方法 “no”：訓練中にloggingを行わない “steps”： loggingをlogging_steps毎に行う “epoch”： 各Epochの終了時にloggingを行う
logging_steps	Int	optional	500	logging_strategy=“steps”とした場合に、loggingを更新するstep数を指定する。

mT5モデルを転移学習させる

BSNアイネット

TransformasのSeq2SeqTrainingArguments（訓練中のチェックポイント/最良モデル保存関連）【発展編】

引数名	型	必須 / 選択	初期設定値 /default	説明
save_strategy	str	optional	“steps”	訓練中のチェックポイント保存方法 “no”：訓練中に保存しない “steps”：save_steps毎に保存する “epoch”：各Epochの終了時に保存する
save_steps	int	optional	500	save_strategy=“steps”とした場合は、この引数でチェックポイントを保存するstepの数を指定する。
save_total_limit	int	optional		値を渡すと output_dirから古いチェックポイントを削除し、チェックポイントの合計数を制限する。Defaultでは上限設定が無いことに注意。
load_best_model_at_end	bool	optional	False	Trueにすると訓練終了時に訓練中に見つかった最良のモデルを読み込む。Trueにする場合、save_strategyとeval_strategyを同じ設定にする必要がある。更に”steps”の場合、 eval_stepsはsave_stepsの倍数に設定しなければならない。
metric_for_best_model	str	optional	説明参照	load_best_model_at_endと組み合わせ、2つの異なるモデルを比較するために使用するメトリックを指定する。”eval_”メトリックである必要はないが、訓練中の評価時に返されるメトリックの名前でなければならない。load_best_model_at_end=Trueかつ指定が無い場合のdefault設定は”loss”（評価損失を使用するため）。なお、この引数に値を設定するとgreater_is_betterがdefaultでTrueとなるため、低い方が良いメトリックの場合はFalseに設定すること。
greater_is_better	bool	optional	説明参照	load_best_model_at_endおよびmetric_for_best_model と組み合わせて使用する。より優れたモデルのメトリックを大きくするかどうかを指定する。Default値は次の通り。 metric_for_best_modelが「loss」または「eval_loss」ではない値に設定されている場合：True metric_for_best_modelが設定されていない場合、または「loss」または「eval_loss」に設定されている場合：False

mT5モデルを転移学習させる

TransformasのSeq2SeqTrainingArguments（seed/optimizer関連）【発展編2】

引数名	型	必須 / 選択	初期設定値 /default	説明
seed	Int	optional	42	訓練の開始時に設定されるrandom seed。実行間での再現性を確保する。model_initにランダムに初期化されたパラメータがある場合は、関数を使用してモデルをインスタンス化する。
data_seed	Int	optional		データサンプラーで使用されるrandom seed。設定されていない場合、データサンプリング用のランダムジェネレータと同じシードを使用する。これはモデルシードに関係なく、データサンプリングの再現性を確保するために使用する。
optim	str training_args.OptimizerNames	optional	“adamw_hf”	使用するオプティマイザ： ”adamw_hf”, ”adamw_torch”, ”adamw_apex_fused”, ”adafactor”
learning_rate	float	optional	5e-5	AdamWオプティマイザの初期学習率（注）
lr_scheduler_type	str SchedulerType		“linear”	使用するスケジューラタイプ： 'linear', 'cosine', 'cosine_with_restarts', 'polynomial', 'constant','constant_with_warmup'

注：ドキュメントでもAdamWオプティマイザの初期学習率と書いてあるが、transformers==4.19.2の段階ではAdaFactorの初期学習率も反映される。
Hugging Face 4.X から Hugging Face 5.0 でAdaFactorに関連する引数の削除と変更が入ると告知されており、将来のVersionでは反映されなくなる可能性がある。

転移学習後のmT5で要約させてみる

テストデータセットの要約文を生成させてCSVファイル形式で出力

入力文字列（元データセットのsummary）：

summarize:ブレグジット（イギリスの欧州連合離脱）をめぐり、イギリスのトニー・ブレア元首相が、2度目の国民投票を求める主張を再び繰り広げている。可能性があるブレグジットの形態1つ1つについて議会で投票し、どれも合意に達しない場合に実施すべきという。

出力文字列（generate）：

【英総選挙2019】 ブレグジットの国民投票向け議論始まる

正解ラベル（元データセットのtitle）：

【検証】 ブレグジットに2度目の国民投票案 実施条件は？ 選択肢は？

生成した要約文を評価する

生成した要約文を評価する

テストデータセットの要約文をどうやって評価するか？

上手く要約できているデータと、できていないデータがある。
133個のテストデータを1個1個見比べる…？

	generated	title	summary
0	【英総選挙2019】 ブレグジットの国民投票向け議論始まる	【検証】 ブレグジットに2度目の国民投票案 実施条件は？ 選択肢は？	ブレグジット（イギリスの欧州連合離脱）をめぐり、イギリスのトニー･ブレア元首相が、2度目の国…
1	【ラグビーW杯】日本大会で準々決勝で4位	【ラグビーW杯】 イングランド、ベスト4に一番乗り オーストラリアを大差で下す	ラグビーワールドカップ（W杯）日本大会は19日、決勝トーナメントが始まり、大分スポーツ公園総…
2	イギリスのワクチン開発、約400人実施へ 新型ウイルスの試験	新型ウイルスワクチン、英大学が臨床試験を開始 300人対象	新型コロナウイルスの新しいワクチンの臨床試験がイギリスで始まった。インペリアル･コレッジ･ロ…
3	トランプ米大統領、金委員長の首脳会談へ 発表で	米朝首脳会談は6月12日にシンガポールで トランプ氏ツイート	ドナルド・トランプ米大統領は10日、北朝鮮の金正恩（キム・ジョンウン）朝鮮労働党委員長との首…
4	【米政権交代】トランプ氏、マテでマティアを指名 米軍司令官指名	【米政権交代】国防長官にマティス退役大将「狂犬」のあだ名も	ドナルド・トランプ次期米大統領は1日、国防長官にジェイムズ・マティス退役大将（66）を指名し…
…	…	…	…

生成した要約文を評価する

要約タスクにおける生成された要約文の評価指標

とても大まかに分けると次のような手法がある

- 複数人の専門家などによる人手での評価手法
- 事前に人間が作成した正解の要約文（参照要約）と、モデルにより生成された要約文間の、一致度や含意関係などの指標を用いた自動評価手法
- 人間が作成した参照要約を使用しない強化学習ベースの自動評価手法

生成した要約文を評価する

要約タスクにおける自動評価指標の代表例

事前に人間が作成した正解の要約文（参照要約）がある場合の自動評価手法例

- BLEU
- ROUGE
- METEOR
- Word Movers Distance
- BERTScore
- MoverScore
- …

注：他にも様々な手法があり、また、今現在も新しい自動評価手法が開発されている。

生成した要約文を評価する

本講義で使用する評価手法：BERTScore

生成された要約文と、予め人間が作成した参照要約文の、意味的な同等性を評価したい

従来手法（BLEUなど）：

表面的な文字の一致度で類似度を計算するため「言い換え表現」に弱い

BERTScore：

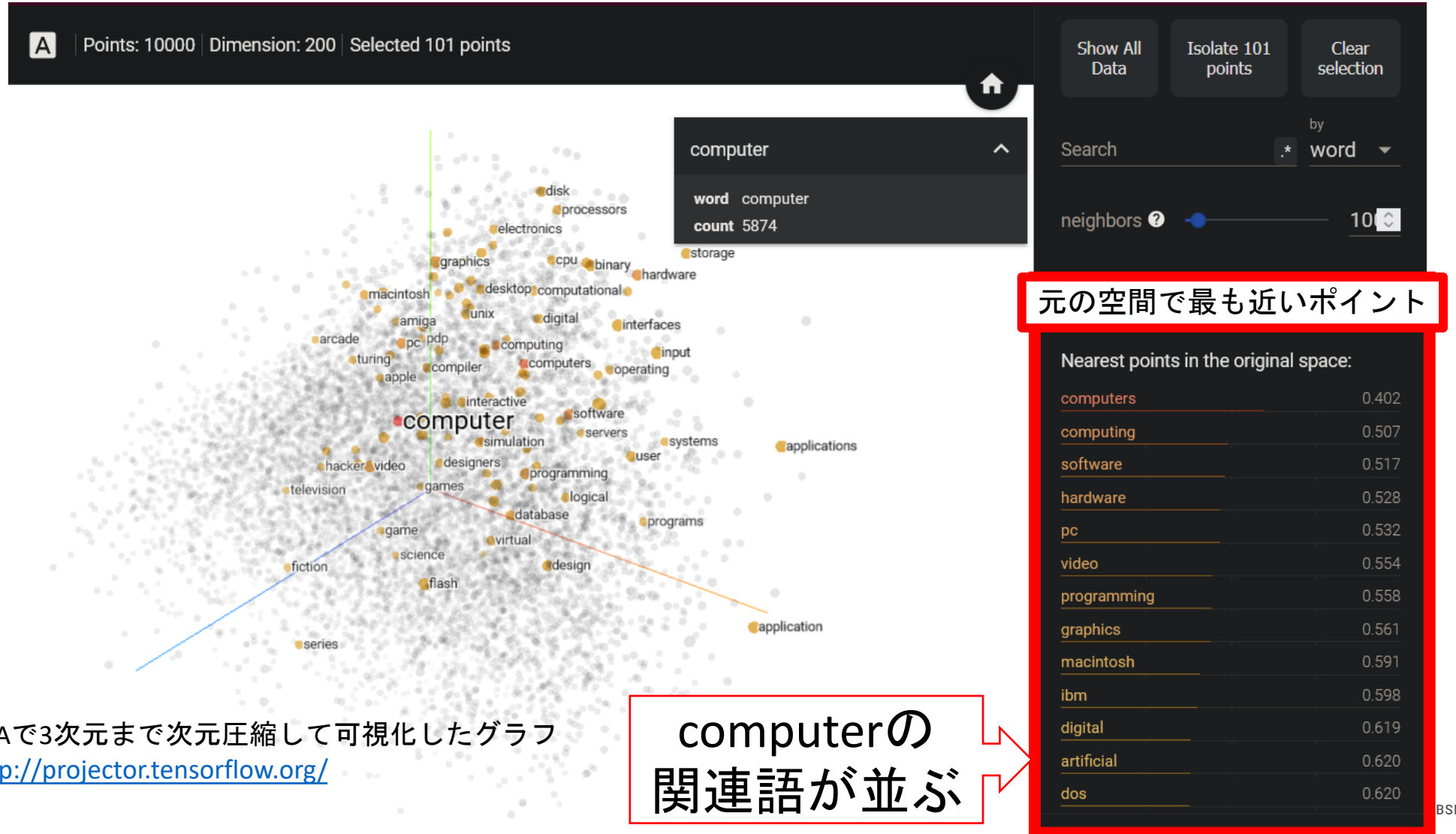
Token Embeddings（ベクトル化されたトークン）間の cos類似度 の合計を類似度として計算

→ 表面的な文字一致よりも言い換え表現に強い

補足；cos類似度 / Cosine Similarity
2つのベクトルがなす角のcos値であり，2つのベクトルの類似性を示す尺度
この値は，-1 から +1 の範囲であり，
+1 ならば なす角が　0度，同じ向きのベクトル　=類似性が高い
0 ならば なす角が　90度，直交ベクトル（独立）=無関係
-1 ならば なす角が 180度，逆向きのベクトル　=類似性が低い
ことを意味する

再掲：単語の分散表現

単語の分散表現：単語を数百次元のベクトルで表現



PCAで3次元まで次元圧縮して可視化したグラフ
http://projector.tensorflow.org/

computerの関連語が並ぶ

生成した要約文を評価する

本講義で使用する評価手法：BERTScore

生成された要約文と、予め人間が作成した参照要約文の、意味的な同等性を評価したい

従来手法（BLEUなど）：

表面的な文字の一致度で類似度を計算するため「言い換え表現」に弱い

BERTScore：

Token Embeddings（ベクトル化されたトークン）間の cos類似度 の合計を類似度として計算

→ 表面的な文字一致よりも言い換え表現に強い

補足；cos類似度 / Cosine Similarity
2つのベクトルがなす角のcos値であり，2つのベクトルの類似性を示す尺度
この値は，-1 から +1 の範囲であり，
+1 ならば なす角が　0度，同じ向きのベクトル　=類似性が高い
0 ならば なす角が　90度，直交ベクトル（独立）=無関係
-1 ならば なす角が 180度，逆向きのベクトル　=類似性が低い
ことを意味する

BERTScore: Evaluating Text Generation with BERT (ICLR 2020)
arXiv : https://arxiv.org/abs/1904.09675

生成した要約文を評価する

BERTScore算出の流れ

文の埋め込み表現
（ベクトル化したトークン）

cos類似度を計算

cos類似度行列を作成し
貪欲マッチ (greedy match)

重要度の重み付け

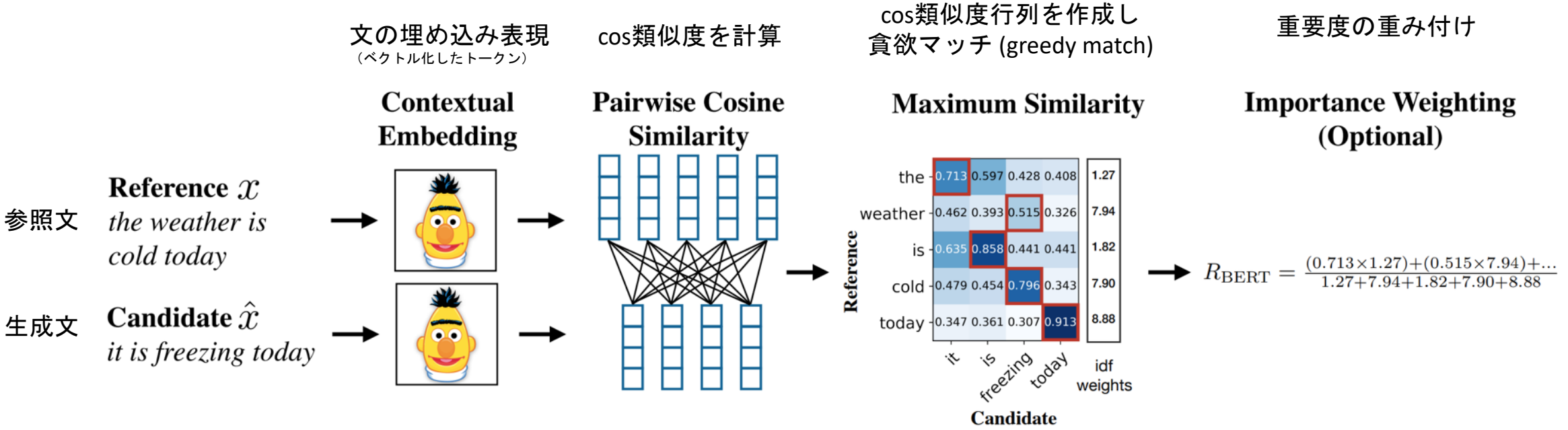


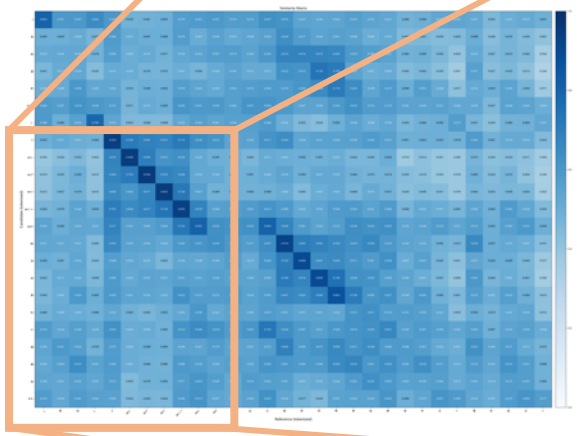
Figure 1: Illustration of the computation of the recall metric R_{BERT}. Given the reference x and candidate $\hat{x}$, we compute BERT embeddings and pairwise cosine similarity. We highlight the greedy matching in red, and include the optional idf importance weighting.

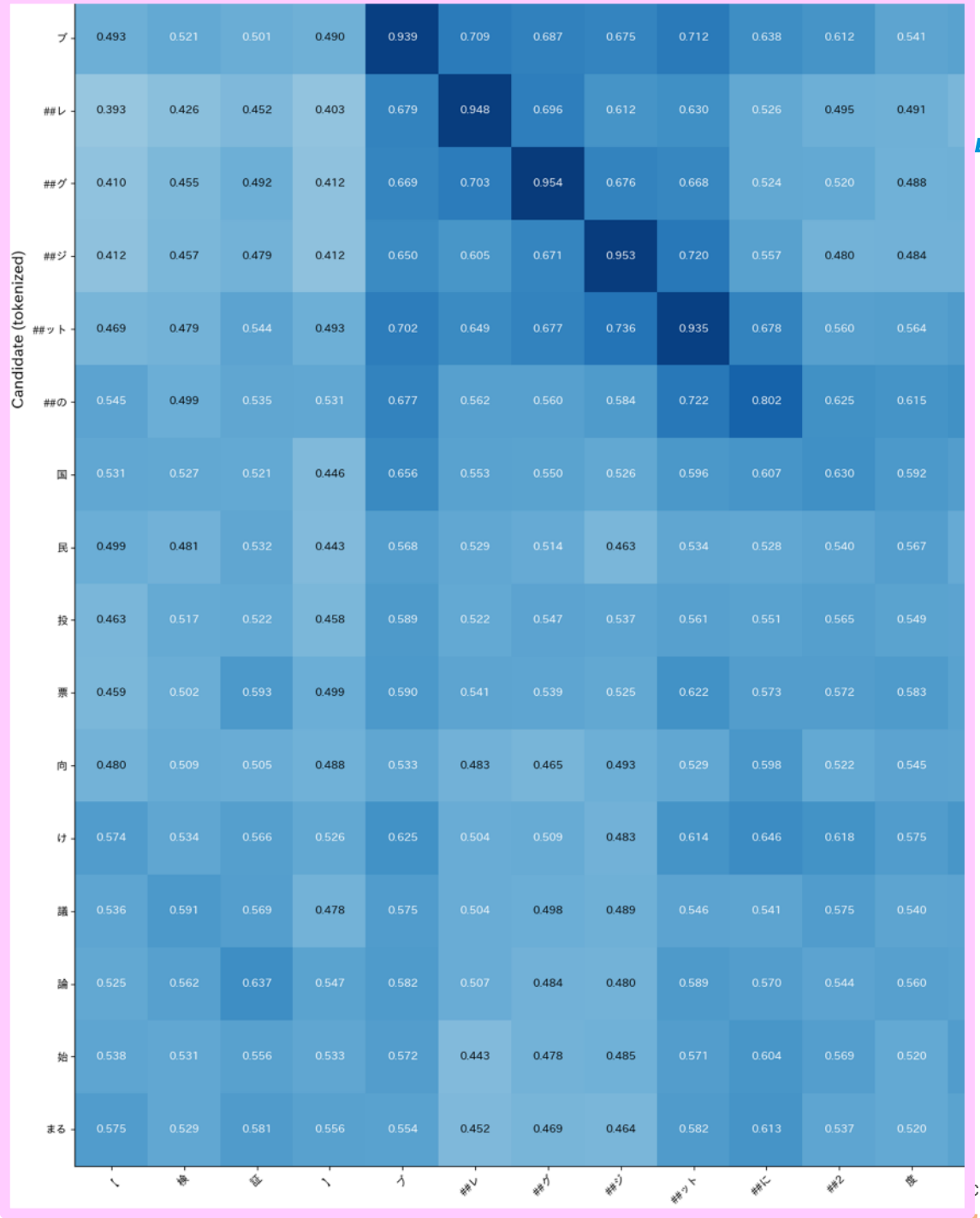
貪欲マッチ （greedy match） ：複数候補の中から最も長い文字列で一致させること

https://arxiv.org/abs/1904.09675

生成した要約文を評価する

cos類似度行列





生成した要約文を評価する

テストデータの要約文をBERTScoreで評価

Precision（適合率；正と予測したものが正だったかの度合い）：

generated（生成要約） の各トークンを、title（正解ラベル/参照要約）の各トークンと照合する

Recall（再現率；真に正であるものが正と予測できたかの度合い）：

title（正解ラベル/参照要約）の各トークンを、generated（生成要約）の各トークンと照合する

F1：

PrecisionとRecallの調和平均

	generated	title	summary	BERTScore_Precision	BERTScore_Recall	BERTScore_F1
0	【英総選挙2019】 ブレグジットの国民投票向け議論始まる	【検証】 ブレグジットに2度目の国民投票案 実施条件は？ 選択肢は？	ブレグジット（イギリスの欧州連合離脱）をめぐり、イギリスのトニー・ブレア元首相が、2度目の国...	0.730920	0.630402	0.676950
1	【ラグビーW杯】日本大会で準々決勝で4位	【ラグビーW杯】 イングランド、ベスト4に一番乗り オーストラリアを大差で下す	ラグビーワールドカップ（W杯）日本大会は19日、決勝トーナメントが始まり、大分スポーツ公園総...	0.804324	0.640319	0.713012
...	...	...	...	...	...	...

補足

https://arxiv.org/abs/1904.09675より定義部分のみ抜粋：

BERTSCORE The complete score matches each token in x to a token in $\hat{x}$ to compute recall, and each token in $\hat{x}$ to a token in x to compute precision. We use greedy matching to maximize the matching similarity score,[2] where each token is matched to the most similar token in the other sentence. We combine precision and recall to compute an F1 measure. For a reference x and candidate $\hat{x}$, the recall, precision, and F1 scores are:

$$R_{\text{BERT}} = \frac{1}{|x|}\sum_{x_i \in x} \max_{\hat{x}_j \in \hat{x}} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \quad , \quad P_{\text{BERT}} = \frac{1}{|\hat{x}|}\sum_{\hat{x}_j \in \hat{x}} \max_{x_i \in x} \mathbf{x}_i^\top \hat{\mathbf{x}}_j \quad , \quad F_{\text{BERT}} = 2\frac{P_{\text{BERT}} \cdot R_{\text{BERT}}}{P_{\text{BERT}} + R_{\text{BERT}}} \quad .$$

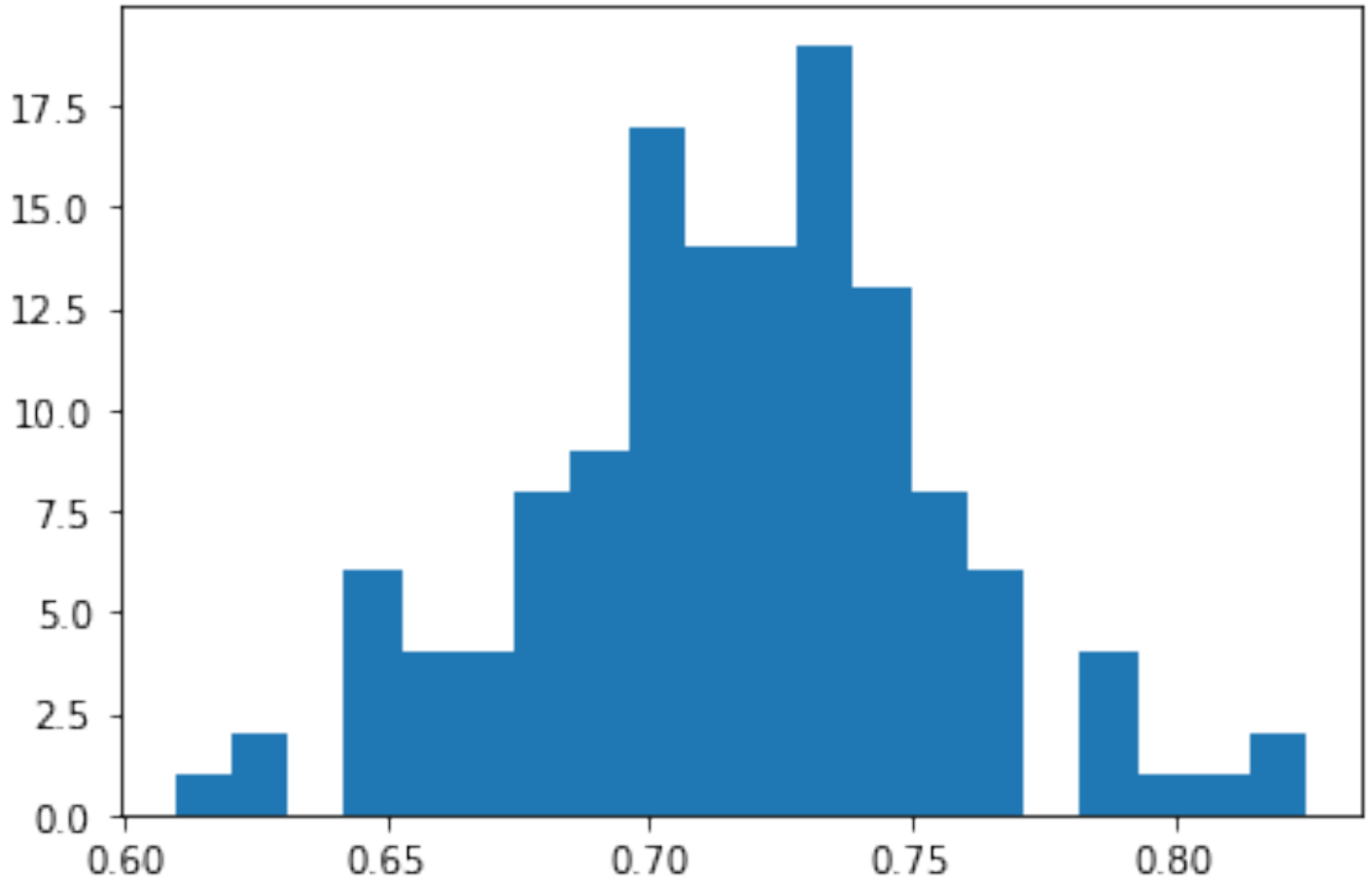
$x = \langle x_1, \ldots, x_k \rangle$: tokenized reference / トークン化された参照要約
$\hat{x} = \langle \hat{x}_1, \ldots, \hat{x}_m \rangle$: tokenized candidate / トークン化された生成要約

$\langle \mathbf{x}_1, \ldots, \mathbf{x}_k \rangle$: 参照要約のトークンが事前学習済BERTによりEmbedding（ベクトル化）されたもの
$\langle \hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_l \rangle$: 生成要約のトークンが事前学習済BERTによりEmbedding（ベクトル化）されたもの

生成した要約文を評価する

テストデータの要約文をBERTScoreで評価

テストデータセット（N=133）のBERTScore_F1分布をヒストグラムで描画



生成した要約文を評価する

BERTScore_F1の最小値と最大値を比較する

	Index	generated	title	summary	BERTScore_Precision	BERTScore_Recall	BERTScore_F1
最小値	105	【解説】 ハッシュタグ「私はこの表情」と知らされず	ハエを叩き落とすのは、なぜこんなに難しいのか？	ハエを叩こうとすると、いかに相手が自分より素早いか実感する。ずっと。しかし、小さいハエの脳み…	0.627673	0.592768	0.609722
最大値	125	米カリフォルニア州で大規模な山火事、100人以上が行方不明と	150人以上が行方不明 カリフォルニア州山火事で	米西海岸カリフォルニア州北部で8日に発生した大規模な山火事で、150人以上が行方不明になって…	0.891646	0.768004	0.825219

引用文献URLや参考資料まとめ

arXiv

[Transformer]

Attention Is All You Need（https://arxiv.org/abs/1706.03762）

[BERT]

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding（https://arxiv.org/abs/1810.04805）

[T5]

Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer（https://arxiv.org/abs/1910.10683）

[mT5]

mT5: A massively multilingual pre-trained text-to-text transformer（https://arxiv.org/abs/2010.11934）

[SentencePiece]

SentencePiece: A simple and language independent subword tokenizer and detokenizer for Neural Text Processing（https://arxiv.org/abs/1808.06226）

[BERTScore]

BERTScore: Evaluating Text Generation with BERT（https://arxiv.org/abs/1904.09675）

引用文献URLや参考資料まとめ

分かりやすく解説しているサイトなど

[日本語]

- 論文解説 Attention Is All You Need (Transformer)（https://deeplearning.hatenablog.com/entry/transformer）
- 深層学習界の大前提Transformerの論文解説！（https://qiita.com/omiita/items/07e69aef6c156d23c538）
- BERT解説：自然言語処理のための最先端言語モデル（https://ainow.ai/2019/05/21/167211/）
- Googleが発表した自然言語処理モデルText-to-Text Transfer Transformer（T5）とは？（https://www.acceluniverse.com/blog/developers/2020/03/Text-to-Text-Transfer-Transformer.html）
- T5（Text-to-Text Transfer Transformer）について少し説明してみる（https://qiita.com/shotasakamoto/items/8d7fcaae24f818e23977）

[English]

- CMU LTI Low Resource NLP Bootcamp 2020（https://github.com/neubig/lowresource-nlp-bootcamp-2020）
- Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network（https://arxiv.org/abs/1808.03314）
- The Illustrated Transformer（https://jalammar.github.io/illustrated-transformer/）
- Illustrated: Self-Attention（https://towardsdatascience.com/illustrated-self-attention-2d627e33b20a）
- BERT Explained: A Complete Guide with Theory and Tutorial（https://medium.com/@samia.khalid/bert-explained-a-complete-guide-with-theory-and-tutorial-3ac9ebc8fa7c）
- Exploring Transfer Learning with T5: the Text-To-Text Transfer Transformer（https://ai.googleblog.com/2020/02/exploring-transfer-learning-with-t5.html）
- T5: a detailed explanation（https://medium.com/analytics-vidhya/t5-a-detailed-explanation-a0ac9bc53e51）

参考資料集

2021-2022年 ：
「分かち書き」からの脱却を目指す動き

【CANINE】【byT5】…

2021年-2022年：「分かち書き」からの脱却を目指す動き

「分かち書き」そのものの問題点

mT5では、mT5Tokenizer（SentencePiece）に学習させた語彙に含まれるsubwordでtokenに分割し、そのtoken IDをmT5モデルに入力している。

1. 使用する文字の種類が多い言語（漢字圏など）では、語彙に収録しきれない。
   - 現実問題として計算量・メモリ・計算時間に制約がある。
   - 語彙に収録するsubwordと文字のバランスを取る必要があり、収録しなかった文字が出現した場合は<unk>となる。
2. 貫通接辞など分割困難な形態論を持つ言語をうまく処理できているとは言い難い。
   - 貫通接辞：語の中の複数の場所に割り込み派生語を作る形態素、セム語派などが該当する。
   - 言語学的に正しい分かち書きを出力できているどうか議論あり
3. 誤字（スペルミス/タイプミス）や翻字など、例外的な文字現象に対して頑健ではない。
   - 翻字：「絵文字」→「emoji」のような別の文字による代用表記
4. トークナイザ毎の互換性…
5. …

2021年-2022年：「分かち書き」からの脱却を目指す動き

CANINE: Pre-training an Efficient Tokenization-Free Encoder for Language Representation

CANINE：言語表現のための効率的なトークン化のないエンコーダの事前学習

- BERTやT5と同様、Transformerのアーキテクチャを利用した事前学習言語モデル

とてもざっくり言えば：

【今まで一般的に使用されている殆どのモデル】明示的にトークン化しての入力が必須

【CANINE】全言語対応かつ分割が不要、文字ベースで動作する

- Unicode文字規格における code point / 符号位置 に変換して入力
- トークンが文字単位になったようなイメージ
- 単語やsubwordのトークンよりも誤字脱字に頑健
- Unicodeの膨大な文字数かつ文字単位のトークンで事前学習を行うための工夫が色々

Jonathan H. Clark, Dan Garrette, Iulia Turc, John Wieting; Canine: Pre-training an Efficient Tokenization-Free Encoder for Language Representation. Transactions of the Association for Computational Linguistics 2022; 10 73–91. doi: https://doi.org/10.1162/tacl_a_00448
[arXiv] https://arxiv.org/abs/2103.06874
[Hugging Face]https://huggingface.co/docs/transformers/model_doc/canine

2021年-2022年：「分かち書き」からの脱却を目指す動き

ByT5: Towards a token-free future with pre-trained byte-to-byte models

ByT5：事前訓練されたバイト単位のモデルでトークンのない未来を目指す

- T5 v1.1 のアーキテクチャを利用した事前学習言語モデル
- 大きく異なる部分はモデルへ入力するときの前処理

とてもざっくり言えば：

【mT5】Sentencepieceで分かち書きしたトークンIDを入力

【ByT5】全言語対応かつ分割が不要、文字ベースで動作する

テキストを UTF-8 で表現されたバイト列として各バイトの値を入力

- https://huggingface.co/docs/transformers/model_doc/byt5
- https://arxiv.org/abs/2105.13626

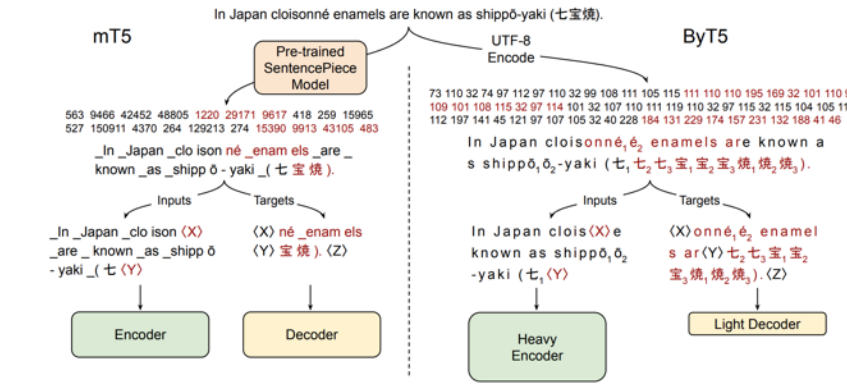


Figure 1: Pre-training example creation and network architecture of mT5 (Xue et al., 2021) vs. ByT5 (this work). **mT5**: Text is split into SentencePiece tokens, spans of ~3 tokens are masked (red), and the encoder/decoder transformer stacks have equal depth. **ByT5**: Text is processed as UTF-8 bytes, spans of ~20 bytes are masked, and the encoder is 3× deeper than the decoder. ⟨X⟩, ⟨Y⟩, and ⟨Z⟩ represent sentinel tokens.

なぜ多言語の処理技術が必要なのか

Digital Divide among Languages / 言語間の情報格差

インターネット上で「**情報を発信**」するまでには；

1. 社会インフラ（インターネットの網）が整っているか
2. 情報端末（コンピュータなど）を利用できるか
3. 情報端末に関する知識
4. 高速なインターネット接続
5. …

インターネット上の言語空間と情報の分裂；

- 全く知らない言語で書かれた情報を読もうとするだろうか
- 全く知らない言語を使用しているコミュニティへ入っていけるだろうか
- インターネット上の言語空間は情報の分裂と情報の不平等へ大きな影響を及ぼす

多言語の処理技術はインターネット上の言語による情報格差を埋める1つの解決策となる

例）自動翻訳など

http://labs.theguardian.com/digital-language-divide/

日本語T5

日本語T5

日本語コーパス（約100GB）で事前学習されたT5 モデル

元々のT5は英語で事前学習されていた
事前学習データ：

- Wikipediaの日本語ダンプデータ （2020年7月6日時点）
- OSCARの日本語コーパス
- CC-100の日本語コーパス

➢ 園部勲さんが作成しHugging FaceとQiitaで情報公開

- [Hugging Face] 日本語T5事前学習済みモデル, https://huggingface.co/sonoisa/t5-base-japanese
- [Qiita] 【日本語モデル付き】2021年に自然言語処理をする人にお勧めしたい事前学習済みモデル, https://qiita.com/sonoisa/items/a9af64ff641f0bbfed44