研究ノート

自然言語分析を用いたメディアバイアスの測定：

地上波放送テレビの字幕データの例[1)]

黒　田　敏　史

要約

本稿では自然言語分析のうち，文書分類タスクにおいて優れた性能を持つとされているBERTを用いたテキストデータの定量分析手法の手続きを紹介する。今回用いる分析用ソフトウェアはHugging Face社によるPython用ライブラリTransformersである。また，分析例としては日本の地情報放送テレビの字幕データを用い，日本の税制に関するメディア報道の傾向を評価する。

1．はじめに

自然言語分析はMikolov et al.（2013）によるword2vec以来顕著な性能向上を遂げた。2022年9月時点では大規模な日本語データを用いて学習した言語モデルが広く公開されており，誰でも手軽に高品質なモデルを用いた自然言語分析ができるようになっている[2)]。栗原他（2022）は日本語言語理解ベンチマークを作成し，文章分類タスク，文ペア分類タスク，QAタスクにおいて東北大BERT, NICTBERT，早稲田大RoBERT, XLM-RoおBERTを比較し，QAタスク以外のタスクにおいては最高得点を得たモデルは人間のスコアと同等以上であるとしている。

Devlin et al.（2018）によるBERTはword2vecの「単語モデル」に，文章の文脈を分析する「言語モデル」を組み合わせたモデルである。word2vecはFirth（1957）による同じ文脈で登場する単語群は近い意味を持っているという仮説から，前後の単語からある単語が登場する確率を最も高く予測する統計モデルを構築し，単語の意味の数量化をするモデルである。「言語モデル」は文章を，文頭の単語が登場する確率，二番目の単語が文頭の単語が登場する条件付きで登場する確率，三番目の単語が一番目と二番目の単語がその順番で登場する条件付き確率，等と文章を単語の順番を踏まえた条件付き確率として捉えるモデルである。これら2つのアプローチを組みあわせ，文中の各単語に前後の文脈を踏まえた「文脈ベクトル」を与える事で，単語が文脈によって異なる意味で用いられる事を踏まえた評価を行う事を現実的な計算量で実現したのがBERTである。

これら自然言語分析手法の改善と，デジタル化により言語データの入手がしやすくなったことから，社会科学においても自然言語をデータとして用いる研究が増加している（Gentzkow et al. 2019）。日本語のテキストデータを用いた自然言語分析を利用した経済分析の例として，文章の近さを用いてメディアの報道のバイアスを評価した Kitamura and Kuroda (2020)，単語ベクトルを財の属性として消費者需要の推定に用いた Kawaguchi et al. (2022) 等が存在している。

本稿の構成は以下のようになる。第二節では日本語のテキストデータを用いて分析するために必要な前処理について紹介する。第三節では Hugging Face 社による Python 用ライブラリ Transformers を用いて BERT を利用する方法を紹介する。第四節では日本の地上波テレビ放送の字幕データを用いた日本の税制に関するメディア報道の傾向の評価を紹介する。

2. 日本語テキストデータの前処理

テキストデータは Python や R 等のコンピュータ言語では文字コードの集合体として記録される。過去には様々な OS や国毎に独自の文字コードが用いられていたが，現代では UTF-8 が国や OS に依存しない文字コードとして広く利用されるようになっており，テキストデータは UTF-8 で扱う事が望ましい。

UFT-8 ではあるコードが特定の一文字を表す事は定まっているが，その文字が中国語の文章として書かれたのか，それとも日本語で書かれてたのかは記録されないため，しばしば言語を判定することが必要となる。テキストデータが日本語であるかを判定する方法として，日本語のみで用いられるひらがな・カタカナを含んでいれば日本語とする手法がある。しかし，例えば Google Play のアプリストアからクロールしたデータでは英語で書かれた説明書きの最後に「日本語には対応していません」という文字列が含まれているような場合があったり，日本語学習ソフトに「こんにちは！」などという文字列が含まれているが，他は全て他言語であったりする場合もあるため，追加的な判断が必要となることもある。Microsoft Azure 等のクラウドサービスの自然言語分析 API には言語判定ツールが含まれている事があるが，これはテキストデータに含まれる当該言語でしか用いられない文字の登場比率等から各言語である蓋然性を計算しているようであり，内容が日本語として理解可能かの判定はしていないようである。

日本語などの東アジア言語は単語間にスペースを入れないため，テキストデータを単語の束に分解することが必要となる。テキストデータを単語の束にするには MeCab[3] 等の形態素解析ソフトウェアを用いる事ができる。MeCab 等のツールは解析アルゴリズムと辞書が独立しており，学習済みモデルを用いる場合は学習データを作成したのと同じ辞書を用いて形態要素解析を行うことが望ましい[4]。また，新たに登場した固有名詞などを単語として認

識するためには Web で使われた単語を自動的に収集し，週 2 回以上辞書を更新している mecab-ipadic-NEologd[5] 等を用いる事が望ましい。mecab-ipadic-NEologd を作成するにあたって用いられた全角半角の統一などの正規化処理を行うための neologdn[6] も配布されており，学習環境と同様の正規化処理を効率的に行う事ができる。しかし，mecab-ipadic-NEologd をインストールするには Linux か MaxOSX 環境が必用であるほか，Google Colab 上で利用するにはさまざまな困難がある。Sudachi[7] は形態要素解析オプションとして UniDi を用いる最小構成から mecab-ipadic-NEologd をベースにした大規模辞書までを選択可能であり，Python ではパッケージ管理ソフトの PIP を利用して容易にインストールできるため，Windows ユーザや Google Colab を用いるユーザは Sudachi を用いても良いだろう。さらに，後述の Hugging Face 社の Transformers では，BERT の学習済みモデルとそれに用いた形態素解析パラメータをセットで配布する事ができるようになっており，利用モデルにあわせて形態素解析環境を整備し，コーディングを行う必要が無くなったことは日本語のテキストデータ分析の研究効率を引き上げている。

多言語を同時に取り扱う言語モデルでは，漢字を 1 つの単語とみなして学習するアプローチが取られることがある。word2vec のような単語が 1 つのみの固有の意味を持つもモデルにおいて日本語と中国語で同じ文字列で表されるが全く異なる意味を持つような場合にこのアプローチは問題が生じるが，コンテキストを踏まえて単語の意味を判断する BERT ではこのアプローチを用いた XLM-RoBERT の性能が他の日本語モデルよりも高く，今後の日本語自然言語分析では単語に拘る意味は余りないかもしれない。

形態要素解析ができればあとは言語モデルに単語の束を評価をさせれば良いが，計算機の保存容量や計算速度が制約となることがある。BERT は 1 つの単語を一層あたり数百次元のベクトルを多層重ねたモデルによって評価する。これは，文字列ではたかだか数バイトの一単語が数キロバイトのデータとして出力され，データ量が最大 1,000 倍以上に膨らむことを意味する。数 MB 程度のテキストデータであっても全テキストデータの評価とその保存にはそれなりの資源を要する事に留意すべきである。また，BERT は word2vec のように単語とベクトルが 1 対 1 対応ではないため，文章に含まれる各単語を評価するタスクにもそれなりの計算時間がかかる。各自の環境で小さなデータを用いて計算時間を確認してから分析に必要なデータの絞り込みをする事が必要となる場合もあるだろう。

3. 日本語 pre-traind BERT の利用

本節では Hugging Face 社による Python 用ライブラリ Transformers を用いて日本語 BERT による自然言語分析を行うための環境構築と，サンプルコードを紹介する。また，本稿で紹介するソフトウェア群は全て大学教員や学生にはフリーで利用可能である。

2022年8月31日時点の「フリーで使える日本語の主な大規模言語モデルまとめ」には日本語BERTモデルとして12の企業・もしくは大学による汎用モデル，10の企業・もしくは大学によるニュース・金融・医療などのドメイン特化型モデルが提供されている。これらのモデルのうち，「HuggingFaceですぐ使える？」とされているものは汎用が6個，ドメイン特化型が4個ある。モデルの選択にあたっては目的に即したモデルを選べば良いが，モデルによててゃ形態素解析ソフトの追加的インストールが必要になったり，予め別のソフトウェアでテキストデータを単語の束に変換しておく事が必要であったりするため，即利用できると言っても一定程度の事前準備は必要である。

ここでは，分析の簡便性から，東北大学乾研究室による「東北大BERT（cl-tohoku/bert-large-japanese)」を用いた事例を紹介する。東北大BERTは栗原他（2022）による性能比較ではそれほど性能が高く無いが，1. Pythonのほかインストールが必要なソフトウェアがMeCabのみである事，2.テキストデータを単語の束にするための処理と，単語の束にBERTモデルで処理するためのindexを与えるtokenizerが一体化しており，テキストデータをそのまま流し込むことができること，3. 追加で導入が必要なライブラリが比較的少ないこと，から導入が容易である。

1）環境構築

本稿ではGoogle Colab環境を用いたケースを紹介する[8)]。ただし，無償のGoogle Colabでは利用できるメモリ量や実行時間の制約から，本格的な分析を行うには様々な困難が生じる。筆者が4節で紹介する分析をした環境はRyzen Threadripper 3770Xにメモリを128GBを積んだWindows10OS搭載のマシンにAnaconda[9)]，MeCabをインストールした環境である。GPUはCUDA非対応のRadeon RX 480を用いているためGPUを用いた計算は行っていない。

Python学習用のIDEにはJupyter Notebook，もしくはJupyterLabが用いられる事が多い。ここでもJupyter NotebookをベースとしたGoogle Colabを用いているが，Jupyter Notebookはあくまでも学習用環境であり，筆者が研究で用いているIDEはPyCharm[10)]である。

2）ライブラリのインストール

東北大BERTを利用するためには，fugashi[11)]，torch[12)]，transformersが必要である。Google Colabであれば，以下のコードをセルにおいて実行することで，これらのライブラリをインストールすることができる。ローカルなAnacondaがインストールされている環境であればanaconda promptにて行頭の!を除いて各行を実行すれば良い。

```
!pip install fugashi[unidic-lite]
!pip install torch
!pip install transformers
```

一つ目の 'fugashi[unidic-lite]' は MeCab を呼び出すためのライブラリと辞書のセットである。GoogleColab であれば MeCab は予めインストールされているが，ローカル環境であれば別途インストールをした上でなければ fugashi は機能しない。二つめの torch は GPU 計算に用いられる tensor 型データを取り扱うためのライブラリである。ただし，GPU が無くても CPU によって計算する事もできる。三つめの transformers は Hugging Face 社による BERT を含んだ自然言語分析用の様々なライブラリを含んだパッケージである。

Hugging Face 社は Hugging Face Hub[13] というオープンソースの自然言語分析モデル・データ・アプリを共有するプラットフォームを設けており，当該プラットフォームに登録されたモデル等は transformers からそれらモデルを初回に呼び出す際に自動的にインストールされる仕組みになっている。東北大 BERT モデル（large）を読み込むためには，

```
bert_model_name ="cl-tohoku/bert-large-japanese"
tokenizer = BertJapaneseTokenizer.from_pretrained(bert_model_name)
model = BertModel. from_pretrained(bert_model_name, output_hidden_states=True)
```

とすれば良い。また，配布されたモデルに含まれていない単語に関心がある場合，単語のリストを作成し，モデルの単語に追加することが可能である。例えば，次節で用いる税制に関する単語を追加したければ

```
tax_keywords = ['消費税', '増税', '消費増税', '軽減税率', '税', '受信料']
tokenizer. add_tokens(tax_keywords)
model. resize_token_embeddings(len(tokenizer))
```

等とすれば良い。

日本語のテキストデータは文章を評価する前に，文字列を単語の束に変換することが必要である。東北大 BERT では BertJapaneseTokenizer が提供されており，これを用いる事でモデル制作者が指定した方法で日本語の文字列を日本語の単語の束にする事ができる。ただし，BERT においては 1 つの文章の始まりを表す "[CLS]"，終わりを表す "[SEP]" を別途要素として与える必要がある。実装例[14] としては以下のようにすれば良い。

```
tokenized_text = tokenizer. tokenize('[CLS]' + ' ここに任意の文章をいれる。' + '[SEP]')
tokenized_text
```

上記の例であれば

```
['[CLS]', 'ここ', 'に', '任意', 'の', '文章', 'を', 'いれ', '## る', '。', '[SEP]']
```

というリストオブジェクトが帰ってくるはずである。'## る' は単語ではなく，単語の構成要素であるサブワードに分解された事を意味する。

BERT による文章評価は単語の束に含まれる各単語を単語インデックスに変換したオブジェクトに変換してから行う必要がある。実装例は以下のようになる。

```
indexed_token = tokenizer. convert_tokens_to_ids(tokenized_text)
indexed_token
```

```
[2, 11962, 893, 15325, 896, 16772, 932, 17661, 6365, 829, 3]
```

BERT モデルは二つのペアとなる文章を用いて学習する。評価においても，最大二つの文章を同時に与える事ができ，モデルに投入された単語インデックスがどちらの文章に属しているかを 0 もしくは 1 の値を取るダミー変数のリストを与えて知らせる必要がある。1 文を評価をするだけであれば 1 を与えればよく，

```
segments_id = [1] * len(tokenized_text)
```

等としてやれば良い。

次に，これらのリスト型式のデータを BERT モデルで利用する tensor 型に変換する。

```
tokens_tensor = torch.tensor([indexed_token])
segments_tensor = torch.tensor([segments_id])
```

以上で BERT で評価するための前処理は終了である。あとは，tokens_tensor と segments_tensor をモデルに与えればモデルによる文章の評価をすることができる。Tensor 型の多次元ベクトルを得るのであれば，

```
with torch.no_grad():
  output = model(tokens_tensor, segments_tensor)
  hidden_state = output[2]
```

等とすれば良い。

hidden_state は先の単語のインデックスと，東北大 BERT の 1 層あたり 768 次元，24 層からなるパラメータとして評価した結果を含んだオブジェクトである。Devlin et al.（2018）によれば，隠れ層の最後の四層のパラメータを 1 次元のベクトルに連結した場合に性能が高いとしているが，1 単語を次元数 * 4 個の値で表すのはデータ容量的に困難な場合もあるだ

ろう。代替的に提案されている手法は，各次元の値を全層や最後の幾つかの層で平均値を取る手法，もしくは最後の層のベクトルである。

hidden_state は文章インデックス（1 個），単語インデックス（トークン数），ベクトル（1024 次元）の要素を持つ各層の tensor が tuple としてまとめられているオブジェクトである。例えば，最後の 4 層の平均を取得するには先の hidden_state から以下のようにしてベクトルを取得すれば良い。

```
layers = 4
token_embedding = torch.stack(hidden_state, dim=0)
token_embedding = torch.squeeze(token_embedding, dim=1)
token_embedding = token_embedding.permute(1, 0, 2)
token_vec = []
for hidden_layers in token_embedding:
    vec = torch.mean(hidden_layers[-layers:], dim=0)
    token_vec.append(vec)
```

一行めで取得する次元数をパラメータとして与え，二行目で各層の tensor からなるタプルを，層を第 1 インデックスに持つ 1 つの tensor に組換え，三行目で文章インデックスを削除し，四行目で tensor のインデックスの並び順を［層］［単語］［次元］から［単語］［層］［次元］に並べ替え，その後単語毎のループ内で後ろから layers 数の層の間で各次元の平均値を計算し，単語毎の意味を表す 1024 次元のテンサーを要素としたリストを構築している。

この単語毎の意味を表す 1024 次元のテンサーのうち，第 1 単語である「[CLS]」は文章全体の意味を表している。また，各単語の 1024 次元のテンサーは各単語が前後の文脈の中で持っている意味を表している。文の類似度比較をする場合，各文の「[CLS]」だけを用いれば良いとされているが，予め似た文章を与えた教師付学習によって既存の BERT モデルをファインチューニングする Sentence BERT（Reimers and Gurevych, 2019）が提案されている。Sentence BERT では「[CLS]」のみならず，文中の単語の加重平均をするモデルが高い性能を持つとしている。同様に，Kawaguchi et al.（2022）では word2vec による単語ベクトルを文章ベクトルに変換する際に，全単語の単純平均よりもカテゴリ内における単語の出現頻度の逆数をウエイトにした全単語の加重平均の方が消費者需要への fit が高かったため，加重平均を用いている。

4. 日本の地上波放送による税制報道の傾向評価

本節では，先の手続きによって取得した文章ベクトルや単語ベクトルを用いて，日本の地

上波放送による税制報道の傾向を評価する。評価を行ったのは，東京広域圏に存在する放送局のうち，NHK 総合 東京，TBS，テレビ朝日，テレビ東京，フジテレビ，日テレである。NHK 教育や MXTV もデータとしては存在するが，NHK 教育は他の放送局とは放送番組が明らかに異なっており比較に適さないこと，MXTV は字幕のカバー率が低いことから分析から除外した。分析期間は 2017 年 1 月 1 日から 2019 年 12 月 31 日までの三年間である。当該期間は 2016 年にリーマンショック級の危機のリスクを認知した故安倍晋三首相が消費税の増税を延期し，その後特段大きな経済ショックはなく，衆議院が解散され選挙が行われ，軽減税率の導入が決定され，増税が実施された時期である。軽減税率は消費税の逆進性を強める効果を持ち，消費を歪ませ，課税コストを引き上げる[15] ため，2019 年 3 月には 100 名を超える経済学者が軽減税率の導入に反対する『軽減税率に関する緊急政策提言』[16] に賛同しており，経済学者からは批判的な見解が出されている。

地上波放送のデータは，2017 年から 2018 年 3 月まではガラポン社の提供するガラポン TV を用いて自宅[17] のある東京都において受診が可能な放送局の字幕を自動取得しており，2019 年についてはガラポン社より全国の地上波放送局のデータを購入したものを用いている。

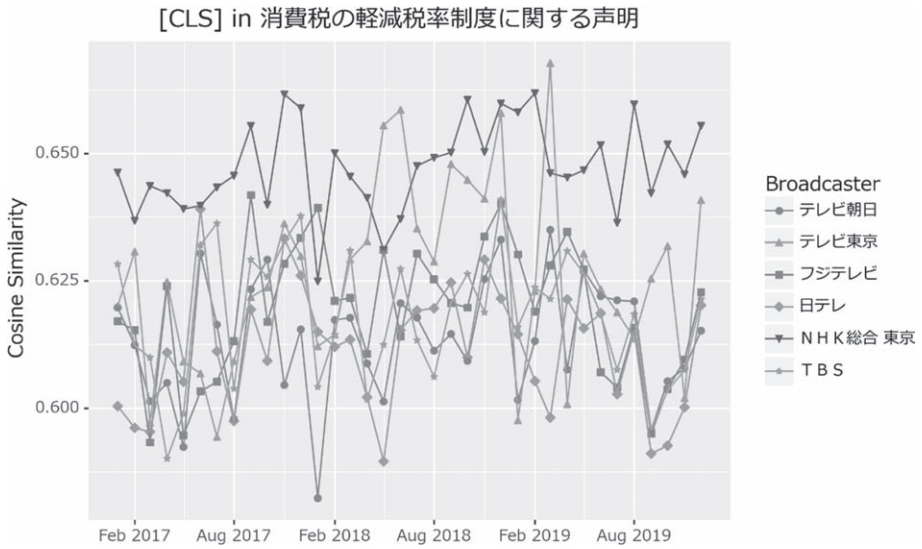
BERT は 1 つの単語を最大数 kb で表すモデルである。先述の 6 つの放送局の 1 日あたりの字幕データは約 25MB である。このデータに含まれる全単語を BERT で評価した結果のデータは，1024 次元の 1 層のベクトルだけであっても 1 日あたり約 17GB のデータ量となり，三年間のデータ量は約 18TB となる。全ての単語の評価結果を取得し，保存するのは筆者の環境では非現実的であったため，1）放送データに含まれるジャンルコードにニュースジャンルを含んでおり，2）字幕データに以下の税制キーワードを含んだ番組のみを分析の対象とした。税制に関するキーワードは，消費税，増税，消費増税，軽減税率，税，受信料である。受信料はこの期間政策的な議論の対象とならなかったために比較用に用いる事とした。これらのキーワードは辞書に含まれているか定かではないため，2 説に記述したとおり単語を辞書に追加してからトークン化と評価を行っている。

また，番組の文字列全てを評価するのではなく，番組全体の文字列を句点や！？等の文末記号毎に文字列を区切って 1 文とみなす変換をした。このうち，税制に関するキーワードの含まれた文のみを抽出し，BERT による評価対象とした。なお，通常の BERT では文脈を考慮できる一文の長さは最大 512 ワードに限られており，512 ワードを超える文章は 512 を超えないワード数に等分割している。

上記の手続きにより残った番組数は 2017 年が 3202 番組，2018 年が 3640 番組，4367 番組である。

BERT による放送局の報道の傾向を評価するため，税制について明確な意向を持って記述された文章との類似度から，報道の傾向を評価する。ここでは軽減税率の導入に賛成する

図 1 「消費税軽減税率の適用にあたっての見解」と各区放送局の報道の近さ



参照文章として，「新聞協会の声明」ページに公開されている「消費税軽減税率の適用にあたっての見解」を用いた。また，軽減税率の導入に反対する参照文章として，NIRA（2016）に含まれる加藤淳子による「軽減税率が招く不公平」を用いた。また，脆性との比較用に用いる受信料についての参照文章として，日本放送協会による「受信料と公共放送についてご理解いただくために」[18] を用いた。

以下の図では，比較対象となる参照文章それぞれの「[CLS]」もしくはキーワードの平均値に対する，上記選抜の結果残った番組に含まれる文章の「[CLS]」もしくはキーワードのコサイン距離を計算し，文章全体，もしくは各キーワードの使われ方の近さを評価した結果を，放送局毎の月間平均値を算出してプロットしたものである。

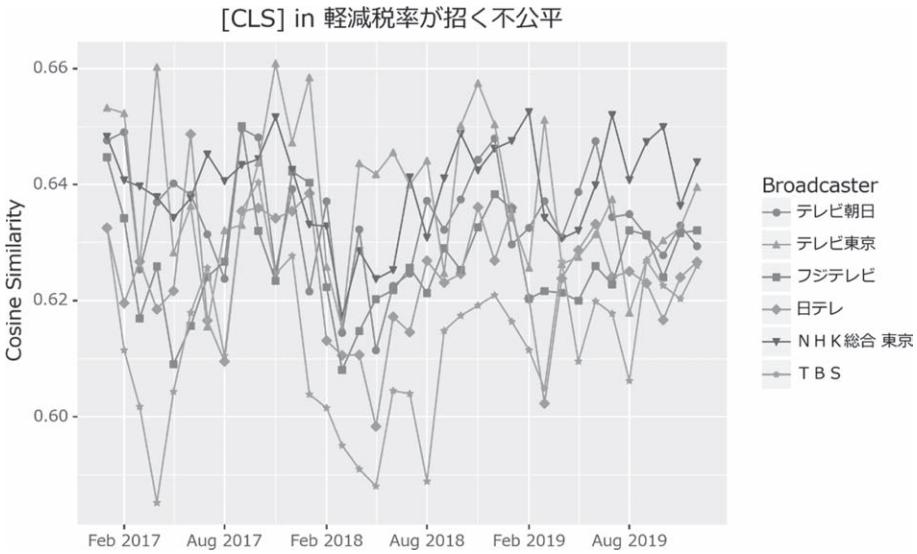
図 1 は 2019 年 10 月 1 日に公開された新聞協会による「消費税軽減税率の適用にあたっての見解」と各放送局の報道の距離をプロットした図である。新聞協会の声明に対し，NHK の報道が他の放送局に比べて近くなっているほか，2018 年の間テレビ東京の報道が新聞協会の声明と近くなっている。また，その他の民放については局内での変動が大きく，放送局間に明確な違いは無さそうである。

図 2 は NIRA で軽減税率反対の巻頭文章である加藤の「軽減税率が招く不公平」と各放送局の報道の距離をプロットした図である。

TBS の報道が他の放送局と比べて加藤と似ていないが，その他の放送局については NHK，テレビ東京，テレビ朝日がやや近く，日テレとフジテレビがやや遠い傾向にあるが，局内の変動も大きい。

軽減税率の導入にあたっては，新聞社を母体とする新聞・テレビ・雑誌に加えインターネ

図２「軽減税率が招く不公平」と各区放送局の報道の近さ



ットサービスでも高い市場シェアを持つメディアコングロマリット企業群が自社グループ企業のために偏向報道をしてたのでは無いかと言説[19]もあるが，データをプロットした限りでは新聞社のグループ企業が加藤よりも新聞協会の声明に近い報道をしているという傾向はない。また，軽減税率導入の影響を受け得ない NHK がいずれの文章に対しても近い報道をしており，特にどちらに近い位報道をしている訳でもない。これらを見ると，テレビの報道は株主や経営者のイデオロギーよりも，消費者の需要に対して動いているとする消費者主権が実現している可能性が示唆される。

続いて，「消費税」についてそれぞれプロットしたのが以下の図である。2017 年初期の水準では各社の「消費税」は新聞協会の用いる「消費税」よりも加藤の用いる「消費税」に近いが，各社が用いる「消費税」と新聞協会の報道の用いる「消費税」の距離が時間と共に近くなっているほか，NHK において特にトレンドの上昇が大きい。一方，加藤の用いる「消費税」と放送局の報道も同様に上昇トレンドがあるが，新聞協会とのそれに比べて購買は小さい。しかし，2019 年時点でも各社の「消費税」の使い方は新聞協会よりも加藤に近い。その他の税制キーワードについても同様のプロットをしたが，得に明確な発見はなかった。また，軽減税率は 2017 年－2018 年中旬までは選挙の期間を除いて殆ど用いられておらず，それ以降も放送で用いられる回数が少ないためか放送局内での分散が大きく，明確な傾向は見て取れなかった。

最後に，「受信料」について，NHK の「受信料と　公共放送についてご理解いただくために」と各社の報道の近さをプロットした。受信料は NHK での登場頻度が高いものの，NHK の使い方と，民放の使い方に著しく違いがある事は無かった。2019 年末に大きく動い

図３「消費税軽減税率の適用にあたっての見解」と各区放送局の消費税の近さ

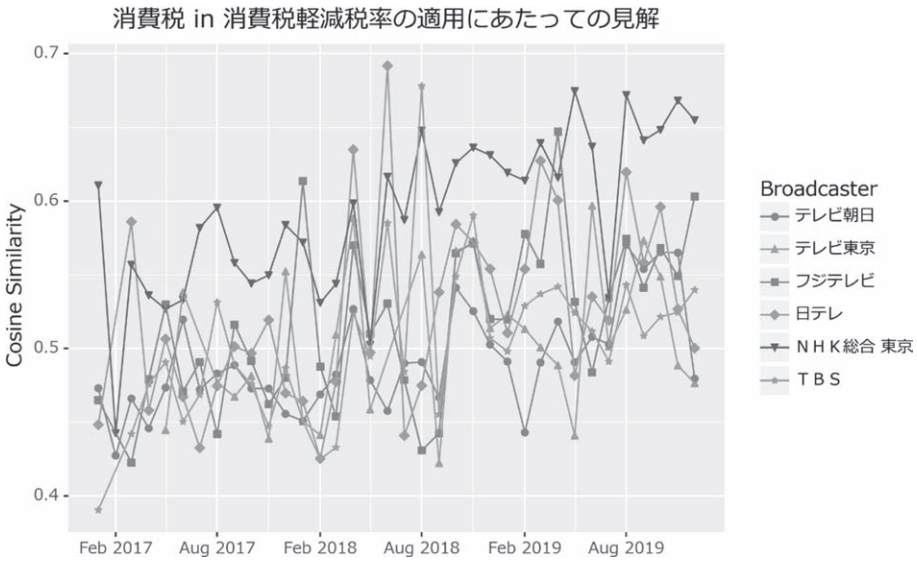
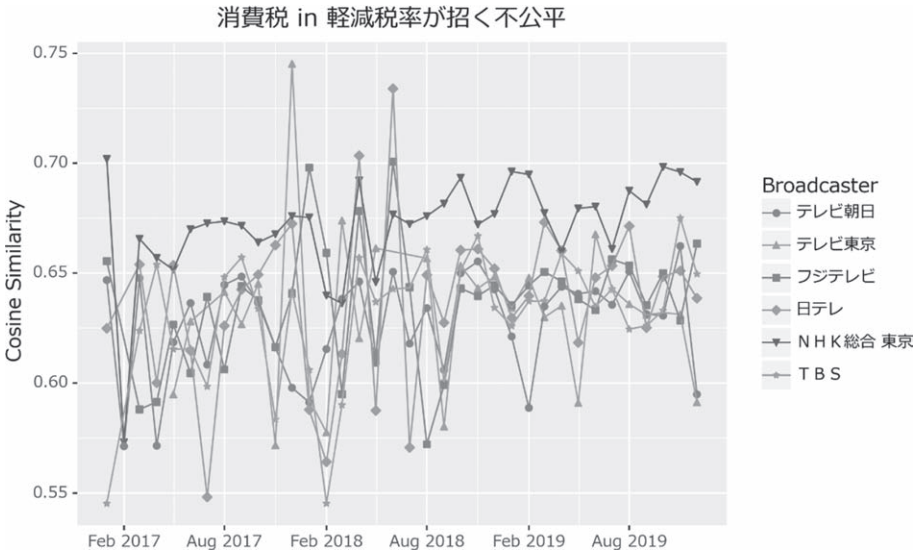


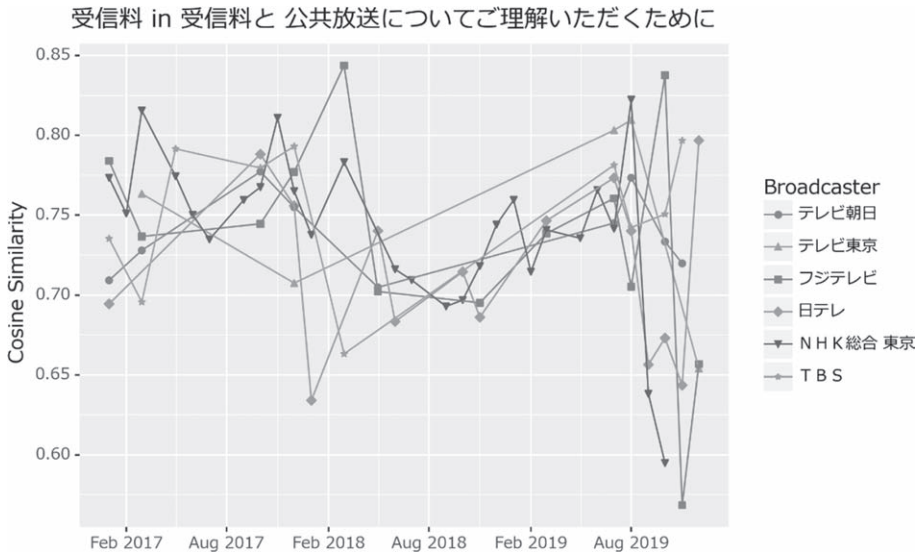
図４「軽減税率が招く不公平」と各区放送局の消費税の近さ



ているのは，2020 年からの受信料引き下げに関する報道が行われたためと考えられるが，NHK 自身が NHK 自身の参照文章から距離を空けた報道をするようになっている。このことも，地上波放送の内容は，株主や経営者の意向よりも，消費者の意向を受けているのではないかとの仮説と整合的である。

以上，BERT モデルを用いて地上波放送の税制についての報道を評価した。仮説検定等

図５「軽減税率が招く不公平」と各区放送局の消費税の近さ



は行っていないが，大まかな推移をプロットする限り，放送局が特定の立場に立って軽減税率について報道していたとする徴候は見られていいない。これが日本の放送法第４条が「政治的に公平であること。」や「意見が対立している問題については，できるだけ多くの角度から論点を明らかにすること。」等としていることが制約となり偏向報道がされていないのか，それとも放送事業者の利潤最大化行動のためなのかは定かではないが，軽減税率がメディアコングロマリットによる放送法違反行為によって導入されたとする明確な傾向は見いだせなかった。本稿はあくまでも BERT の利用方法の紹介であり，同課題を検証するための頑健な統計分析は今後の課題である。

注

１）本研究は，2021 年度の東京経済大学個人研究助成費（研究番号 21－10）を受けた研究成果である。

２）無償で利用可能な日本語言語モデルについては「フリーで使える日本語の主な大規模言語モデルまとめ」https://zenn.dev/hellorusk/articles/ddee520a5e4318#fn-4d2d-1（2022 年 9 月 5 日アクセス）が詳しい。

３）https://taku910.github.io/mecab/（2022 年 9 月 5 日アクセス）はオープンソースの形態素解析エンジンであり，Python や R 等の言語から MeCab を利用するツールなどが豊富であるためよく利用されてきた。

４）公開されている日本語 BERT モデルの形態要素解析ソフトウェアとアルゴリズムの組みあわせについては “awesome-bert-japanese” https://github.com/himkt/awesome-bert-japanese（2022 年 9 月 6 日アクセス）に纏められている。

5）https://github.com/neologd/mecab-ipadic-neologd（2022 年 9 月 5 日アクセス）
6）https://pypi.org/project/neologdn/（2022 年 9 月 5 日アクセス）
7）https://github.com/WorksApplications/Sudachi（2022 年 9 月 5 日アクセス）
8）本稿で用いたコードは以下の URL にて公開している。https://colab.research.google.com/drive/1yxZ7yMiIF4EhOhZV7fUomP0Pywwg7JEg? usp=sharing
9）Anaconda（https://anaconda.org/，2022 年 9 月 6 日アクセス）はデータサイエンス向けに Python 本体，並びに各種ライブラリをパッケージ化したものである。プロジェクト毎に Python やライブラリのバージョンを管理できる Conda 仮想環境は自然言語分析プロジェクトにおいて遭遇しがちなライブラリのバージョンアップによる再現性喪失の防止に有益なため，ローカルな環境に Python を導入する場合は Anaconda の利用が望ましい。
10）PyCharm（https://www.jetbrains.com/pycharm/，2022 年 9 月 7 日アクセス）は Python のみならず，R や Julia 等の言語が混在した PJ を管理する上で得に有益である。
11）fugashi（https://github.com/polm/fugashi，2022 年 9 月 6 日アクセス）は Python から MeCab を利用するためのライブラリであり，東北大 BERT の tokenizer は fugashi を用いて文字列を単語の束にする事ができる。また，辞書として unidic を利用する場合はライブラリと辞書を合わせてインストールすることができる。
12）BERT は GPU で処理することを想定された tensors 型のデータの計算によって実装されており，GPU が利用可能であることが望ましい。一方，CPU によって計算することも可能であり，いずれを利用するかは各自の環境に合わせた torch をインストールする必要がある。下記の例は Windows 上で CPU 計算を利用する場合のインストールコマンドであるが，https://pytorch.org/get-started/locally/ にあるインタラクティブテーブルからビルド，OS，パッケージ管理ツール，利用言語，GPU か CPU か等に合わせたインストールコマンドを取得することができる。
13）https://huggingface.co/docs/hub/index（2022 年 9 月 6 日アクセス）
14）以下の実装例は Chris McCormick 氏による "BERT Word Embeddings Tutorial" を参考にした。https://mccormickml.com/2019/05/14/BERT-word-embeddings-tutorial/#3-extracting-embeddings（2022 年 9 月 6 日アクセス）
15）NIRA（2016）等を参照せよ。
16）https://sites.google.com/view/suggestiondifftaxpublic/（2022 年 9 月 6 日アクセス）
17）筆者の大学研究室にはテレビ放送のアンテナ線配線が存在せず，室内アンテナにブースターを接続しても十分な受信感度が得られなかったためにやむを得ず自宅に機器を設置し，日本放送協会と受信契約を結んだ。
18）https://www.nhk-cs.jp/jushinryo/pdf/jushinryoandkoukyouhousou.pdf（2022 年 9 月 6 日　アクセス）
19）https://omoson.com/tv/keigen/（2022 年 9 月 6 日アクセス）等を見よ。

参 考 文 献

Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. “Efficient Estimation of Word Representations in Vector Space.” *1st International Conference on Learning Representations,*

ICLR 2013 - Workshop Track Proceedings, 1–12.

栗原健太郎，河原大輔，and 柴田知秀．2022.“Jglue：日本語言語理解ベンチマーク．”*言語処理学会 第 28 回年次大会* 0：2023–28.

Devlin, Jacob, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. “BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding.” *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference* 1（Mlm）: 4171–86.

Firth, J R. 1957. *Papers in Linguistics, 1934–1951*. London: Oxford University Press.

Gentzkow, Matthew, Bryan Kelly, and Matt Taddy. 2019. “Text as Data.” *Journal of Economic Literature* 57（3）: 535–74.

Kitamura, Shuhei, and Toshifumi Kuroda. 2022. “Media Trust and Persuasion.” *SSRN Electronic Journal*.

Kawaguchi, Kohei, Toshifumi Kuroda, and Susumu Sato. 2022. “Merger Analysis in the App Economy: An Empirical Model of Ad-Sponsored Media.” *SSRN Electronic Journal*.

Reimers, Nils, and Iryna Gurevych. 2019. “Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks.” *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*, 3982–92.

NIRA 総研．2020.“My Vision デザイン思考で人間中心の政策を．”わたしの構想 46（26）.