一般社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
IEICE Technical Report
KBSE2020-38(2021-03)

BERTによるアプリケーションレビュー分類モデルの構築と評価

山田　侑樹†　　櫨山　淳雄†

† 東京学芸大学 〒184-8501 東京都小金井市貫井北町 4-1-1

あらまし　ソフトウェアの進化の過程でエンドユーザーの意見やフィードバックは開発者にとって重要である．これらの情報を開発者が効率よく利用するために自然言語処理と機械学習を用いた方法が報告されている．本研究ではアプリケーションレビューを分類する機械学習モデルの構築に取り組む．本研究では．これまで報告されている Bag of Words とナイーブベイズ，fastText とロジスティック回帰による分類モデルに加え，BERT による分類モデルを構築し性能評価を実施した．BERT は近年，自然言語処理システムの性能を飛躍的に向上させた機械学習アルゴリズムである．その結果，BERT による分類モデルが Precision 0.7237，Recall 0.7286，F1-Score 0.7173 と最も高い性能を示した．

キーワード　BERT，機械学習，自然言語処理，アプリケーションレビュー

Building of an Application Reviews Classifier by BERT and Its Evaluation

Yuki YAMADA† and Atsuo HAZEYAMA†

† Tokyo Gakugei University, 4-1-1 Nukuikita-machi, Koganei-shi, Tokyo 184-8501 JAPAN

Abstract　In the process of software development, feedbacks from users are important for developers. Most of this information takes the form of a text. In this research, we build and evaluate a machine learning model to classify the application reviews. In addition to classification models based on Bag of Words and Naive Bayes, fastText and logistic regression, we constructed and evaluated a classification model based on BERT. BERT is a machine learning algorithm that has dramatically improved the performance of natural language processing systems in recent years. The results show that the BERT classification model has the highest performance with Precision 0.7237, Recall 0.7286 and F1-Score 0.7173.

Key words　BERT, Machine Learning, NLP, Application Reviews

1. ま え が き

ソフトウェアの進化の過程でエンドユーザーの意見やフィードバックは開発者にとって重要な情報源である [1]．開発者はユーザーからのソフトウェアの使用感や新たな要望などの情報を得ることによって，ソフトウェアを改善し，より品質の高いものとして提供することができる．例えばアプリケーションレビューでは開発者がリリースしたアプリケーションに対しエンドユーザーは使用感や意見，感想を自由に書き込むことができる．これらの情報はテキスト形式であり，活用するためには人的コストをかける必要がある．そこで，これらの情報を開発者が効率よく利用するために自然言語処理と機械学習を用いた方法が報告されている．

本研究では，このアプリケーションレビューの教師あり学習による自動分類手法に着目する．アプリケーションレビューを対象とした教師あり学習での既存研究では特徴量に Bag of Words（BoW）や fastText を用いたものが報告されている [2] [3]．本研究では既存手法に加え，近年，自然言語処理システムの性能を飛躍的に向上させた BERT による機械学習モデルを構築し性能評価を実施した．本論文ではその結果を報告する．

2. 本研究のアプローチ

本研究は図 1 の手順にしたがって実験を実施する．手順は 1〜5 までである．手順 1 では実験に使用するアプリケーションレビューの取得などデータセットの準備を行う．手順 2 では手順 1 で準備したデータセットに対しノイズとなるデータの除去と前処理を実施する．手順 3 では前処理を施したデータから機械学習アルゴリズムで使用する特徴量の抽出を行う．手順 4 と手順 5 では機械学習アルゴリズムの訓練と評価を実施する．なお本実験では機械学習アルゴリズムに Naive Bayes，ロジスティック回帰，BERT の 3 つを使用しそれぞれの性能を比較する．

以下ではそれぞれの手順で実施する具体的な内容について

述べる．

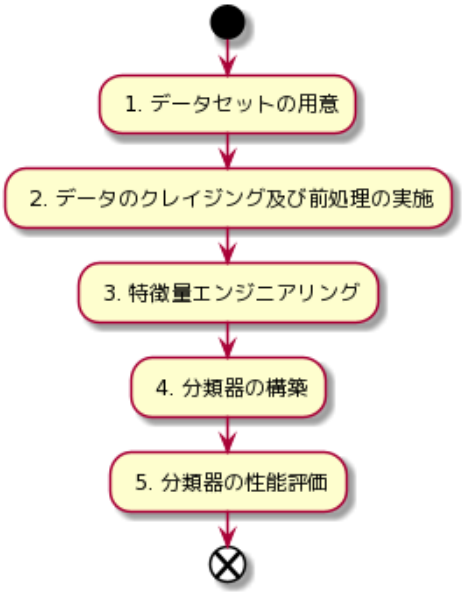

図 1　本研究の手順

2.1　使用するデータセット

本研究では文献 [2] で使用しているアプリケーションレビューのデータセットに加え，新規に取得したアプリケーションレビューに対しラベル付を実施したデータを用いる．新規に取得したアプリケーションレビューには App Store の「仕事効率化」カテゴリの中で上位にあった Keynote(注1) アプリのレビューを用いた．アプリケーションレビューの取得には iTunes Search API (注2) を利用し，500 件のレビューを取得した．取得した 500 件のアプリケーションレビューは文献 [2] のコーディングガイドラインにしたがって著者が手動でラベル付を実施した．

データセットに含まれる各アプリケーションレビューには Bug reports，Feature requests, User experiences, Ratings の 4 種類のいずれかのラベルが付与されている．4 種類のラベルの説明を表 1 に示す．

表 1　4 種類のラベルの説明

ラベル名	概要
Bug reports	アプリの動作や特定の機能の不具合などの問題を報告するレビュー
Feature requests	アプリの特定の既存の機能の使用感などに言及したレビュー
User experiences	機能や機能の欠落，コンテンツの欠落，実装や改善すべき機能などに言及したレビュー
Ratings	アプリ内の機能を褒めたレビュー

2.2　前　処　理

次にデータセットからノイズとなるデータの除去と前処理を実施した．実施した手順は以下の通りである．

（1） レビュー本文が英語ではないデータをノイズとし，除去する．

（2） レビュータイトルとレビュー本文がともに欠損値であるレビューを削除する．

（3） Stopwords を適用することを考慮して NLTK(注3) の英語の Stopwords を適用した際に文章の長さが 0 となるレビューを削除する．

（4） データセットのタイトルとレビュー本文をスペース区切りで 1 つの文章とした．

（5） 文章の全文を小文字に統一し特殊文字の除去を行った．

この結果，データセットからは 4,503 件のデータを得た．データセットに含まれる各レビュー文の割合とレビュー文に含まれる平均の単語数を表 2 に示す．

表 2　データセット中の各ラベルの割合と平均単語数

ラベル	データ数	平均単語数
Bug reports	416	43.8
Feature requests	513	50.3
User experiences	783	37.3
Ratings	2,791	17.9
合計	4,503	27.3

2.3　特徴量エンジニアリング

前処理を実施して得られた文章から自然言語処理による特徴量エンジニアリングを実施し，特徴量を作成した．

2.3.1　Bag of Words による特徴量

Bag of Words（BoW）は文章内の単語の出現頻度に着目するカウントベースの特徴量である．BoW では全文書に出現する全ての単語の辞書を作成し，各単語が文書に出現する回数を数える単純な特徴量である．例えば，“This is an apple. I eat this apple.” という文から BoW によって特徴量を得ることを考える．出現する全単語は [“this”, “is”, “an”, “apple”, “i”, “eat”] の 6 単語である．これらをもとに元文章である “This is an apple. I eat this apple.” で各単語が出現した回数をカウントし，文章を [2, 1, 1, 2, 1, 1] とベクトル表現にすることができる．

2.3.2　fastText による特徴量

fastText は BoW とは異なり，ニューラルネットワークに基づく文章内の単語の推論によって得られる特徴量である．fastText はサブワードモデルを組み込んでいる．サブワードとは 1 つの単語を分割された単語の和とする概念である．fastText では 1 つの単語を 3〜6 文字の N-gram に分割する．例として英語の where という単語を取り上げる．fastText では単語の開始と終了を示す記号として単語のはじめとおわりにそれぞれ，<, >を入れる．すると where は <where>となる．<where>の N-gram を $n = 3 \cdots 6$ の場合で考えると以下のようになる．

- $n = 3$: <wh, whe, her, ere, re>

(注1)：Keynote, https://apps.apple.com/us/app/keynote/id409183694
(注2)：iTunes Search API, https://developerapple.com/library/archive/documentation/AudioVideo/Conceptual/iTuneSearchAPI/
(注3)：NLTK, https://www.nltk.org/

表 3　実験条件と表記

表記名	概要
default	前処理を施した文章をそのまま使用する．
stopwords	前処理を施した文章から Stopwords に含まれる単語を除去して使用する．
lemmatization	前処理を施した文章に各単語を見出し語化する処理を施して使用する．
lemmatization + stopwords	前処理を施した文章の各単語を見出し語化する処理を施し，さらに文章から Stopwords に含まれる単語を除去して使用する．

- $n = 4$: <whe, wher, here, ere>
- $n = 5$: <wher, where, here>
- $n = 6$: <where, where>

よって <where>は 14 通りのサブワードに分割され，それぞれのサブワードの分散表現の和として表現される．このサブワードによって fastText では未知語に対しても単語の分散表現を得ることが可能となる．

本研究では Wikipedia，UMBC webbase corpus，statmt.org news dataset によって事前学習済みの fastText モデル(注4)を使用する．この単語ベクトルの次元は 300 次元である．fastText で得られるのは各単語の分散表現であるため，本実験ではレビュー文に含まれる全ての単語から得られる分散表現を平均化したものをレビュー文の特徴量として扱う．

2.3.3　特徴量エンジニアリング実施時の条件

BoW，fastText によって特徴量を得る際に実験条件として，前処理を施した文章をそのまま用いる場合（default）と Stopwords を適用した場合（stopwords），レビュー文に含まれる各単語の見出し語化を適用した場合（lemmatization），Stopowrds と見出し語化の両方を適用した場合（lemmatization + stopwords）の 4 つを設定する．実験条件をまとめると表 3 のようになる．

2.4　使用するアルゴリズム

本実験で構築する分類器は 4 種類の出力を持つ多クラス分類器である．分類器の構築に Naive Bayes，ロジスティック回帰，BERT の 3 つのアルゴリズムを用いる．Naive Bayes では BoW を，ロジスティック回帰では fastText を，BERT には前処理後の文章を入力値として用いた．

2.4.1　Naive Bayes

Naive Bayes はベイズの定理を応用した手法である．Naive Bayes はテキスト分類で使用される単純な機械学習アルゴリズムであり，特徴量の独立の仮定から BoW と非常に相性が良いとされる．Naive Bayes において未知語の扱いを決定するパラメータ α の値は $\alpha = 1$ とした．

(注4)：fastText - English word vectors, https://fasttext.cc/docs/en/english-vectors.html

2.4.2　ロジスティック回帰

ロジスティック回帰は，二値分類器のための線形モデルであるが一対他 (One-vs-Rest) に拡張することで多クラス分類器に拡張可能である．ロジスティック回帰における罰則項の強さを決めるパラメータ C の値は $C = 50$ とした．

2.4.3　BERT

BERT は 2018 年の後半に Google から発表された自然言語処理を行うための深層学習モデルである [4]．BERT は Bidirectional Encoder Representations from Transformers の略であり Transformer がベースとなっている．Transformer は文献 [5] で発表された Attention メカニズムを活用した深層学習モデルである．

BERT は転移学習によりさまざまなタスクに対応でき，また少ないデータを追加で学習することによってモデルの構築を行える．本実験では BooksCorpus と Wikipedia によって事前学習された 12 段の BERTLayer を持つ BERT-Base モデルを最終段の 12 段目のみ fine tuning を実施して使用する．

BERT での学習を評価データと独立して実施するために訓練データを 8:2 の比率で BERT 用の訓練データと検証データに分割して実施した．なおバッチサイズを 32，最大エポック数を 30，損失関数には交差エントロピー，最適化手法には Adam，学習率を $3e-5$ として学習を実施した．

2.5　モデルの構築および評価方法

モデルの構築と評価は，K-Fold cross-validation（K 層交差検証）に基づいて行う．本研究では $K = 5$ として，データセットを 4:1 の訓練データと評価データに分割し 5 通りの学習と評価を実施する．

性能評価指標には，Precision（精度），Recall（再現率），F1-Score を用いる．なお表 2 に示すようにデータセットに含まれる各ラベルには偏りがあるため各ラベルに含まれるサンプル数で重み付けを行った．

3.　結　　果

BoW を特徴量とした Naive Bayes による分類器の各条件での実験結果を表 4 に示す．BoW を特徴量とした Naive Bayes による分類器では lemmatization + stopwords の条件で構築した際に最も高い精度 Precision 0.6310，Recall 0.6596，F1-score 0.6396 を示した．この lemmatization + stopwords の条件で構築された BoW を特徴量とした Naive Bayes による分類器の性能の詳細を表 5 に示す．

次に fastText を特徴量としたロジスティック回帰での分類器の各条件での実験結果を表 6 に示す．fastText を特徴量としたロジスティック回帰での分類器では stopwords の条件で構築した際に最も高い精度 Precision 0.6336，Recall 0.6704，F1-score 0.6359 を示した．この stopwords の条件で構築された fastText を特徴量としたロジスティック回帰による分類器の性能の詳細を表 7 に示す．

BERT による分類器の実験結果を表 8 に示す．BERT による分類器では Naive Bayes，ロジスティック回帰による分類器のどちらよりも高い性能 Precision 0.7237，Recall 0.7286，

F1-Score 0.7173 を示した．BERT での学習の様子を図 2 に示す．図は $K = 1$ での学習の様子である，図 2 の左がエポックごとの訓練データ（train）と検証データ（val）に対する正解率（accuracy）の推移で，図 2 の右がエポックごとの訓練データ（train）と検証データ（val）に対する損失（loss）の推移である．各ループで同様な傾向が見られ，おおよそ 13 エポックで early stopping を実施し取り出したモデルを評価に用いた．

表 4　Naive Bayes による分類器の各条件での実験結果

条件	Precision	Recall	F1-score
default	0.6270	0.6565	0.6355
stopwords	0.6235	0.6518	0.6322
lemmatization	0.6299	0.6580	0.6379
lemmatization + stopwords	**0.6310**	**0.6596**	**0.6396**

表 5　Naive Bayes による分類器の実験結果 (条件:lemmatization + stopwords)

ラベル名	Precision	Recall	F1-score	sample
Bug reports	0.5426	0.4399	0.4829	416
Feature requests	0.5469	0.5166	0.5277	513
User experiences	0.3264	0.2133	0.2564	783
Ratings	0.7451	0.8438	0.7911	2791
total	0.6310	0.6596	0.6396	4,503

表 6　ロジスティック回帰による分類器の各条件での実験結果

条件	Precision	Recall	F1-score
default	0.6312	0.6684	0.6259
stopwords	**0.6336**	**0.6704**	**0.6359**
lemmatization	0.6280	0.6660	0.6247
lemmatization + stopwords	0.6259	0.6642	0.6275

表 7　ロジスティック回帰による分類器の実験結果 (条件:stopwords)

ラベル名	Precision	Recall	F1-score	sample
Bug reports	0.5615	0.2861	0.3779	416
Feature requests	0.5533	0.5068	0.5281	513
User experiences	0.3947	0.1928	0.2574	783
Ratings	0.7261	0.8918	0.8003	2,791
total	0.6336	0.6704	0.6359	4,503

表 8　BERT による分類器の実験結果

ラベル名	Precision	Recall	F1-score	sample
Bug reports	0.6231	0.6611	0.6410	416
Feature requests	0.7018	0.5965	0.6438	513
User experiences	0.5198	0.3806	0.4144	783
Ratings	0.7999	0.8606	0.8271	2,791
total	0.7237	0.7286	0.7173	4,503

4. 考　　察

本節では前節の実験結果をもとに考察を述べる．

4.1 各アルゴリズムで構築した分類器の混同行列による比較

本実験の結果の F1-score で比較すると BERT, Naive Bayes, ロジスティック回帰の順で高い性能を示した．これについて各分類器の混同行列を作成し比較する．各分類器の混同行列を図 3 に示す．図 3 は左から順に lemmatization + stopwords の条件で構築した Naive Bayes による分類器，stopwords の条件で構築したロジスティック回帰による分類器，BERT による分類器である．混同行列の作成には各分類器の最終（$K = 5$）のループで構築した分類器と評価データを用いている．混同行列を確認すると Naive Bayes，ロジスティック回帰の 2 つの分類器では Ratings と予測することが多いことが伺える．BERT での分類器は他の 2 つの分類器と比較して明らかに各ラベルを学習できていることがわかる．一方で BERT では User experiences と Ratings での誤判定が多いことがわかる．これは，この 2 つのラベルに含まれるレビュー文の類似性が示唆される．

4.2 Naive Bayes を用いた分類器の性能について

文献 [2] で報告されている Naive Bayes と BoW を用いた多クラス分類器の性能は F1-Score が 0.50 であるのに対し，本実験では Naive Bayes を用いた分類器の lemmatization + stopwords の条件で構築した際の F1-score は 0.6396 と大きく向上している．これは文献 [2] では実施されていなかったデータセットからノイズとなるデータの除去や前処理が影響していると考えられる（本実験の手順 2）．また Naive Bayes において lemmatization + stopwords の条件で最も高い性能を示したことについては BoW での特徴量が単語の出現頻度を利用することからも納得のいく結果であった．

4.3 ロジスティック回帰を用いた分類器の性能について

fastText を特徴量としたロジスティック回帰での分類器では単語の見出し語化（lemmatization）の処理を加えると性能の低下が見られた．これは fastText のサブワードの特性を生かしきれず，各ラベルの学習が進まなかったためだと考えられる．

4.4 BERT で構築した分類器の Attention の可視化

BERT では Attention メカニズムを利用している．これを利用して予測結果に対し強く Attention をかけた単語の可視化を行った．この可視化の例を図 4 に示す．図 4 の例では “ipad” や “fix” の単語に強く Attention がかかっていることがわかる他に “there is no way to move edited photos” というバグの本質的な情報を示す部分にも Attention がかかっていることがわかる．

5. 妥当性への脅威

本節では実施した実験結果の妥当性への脅威について述べる．

5.1 データセットのラベル付について

まずデータセットのラベル付が分類器の性能に影響している可能性が考えられる．本実験では文献 [2] で使用しているデータセットに加え，追加でアプリケーションレビューを取得し手

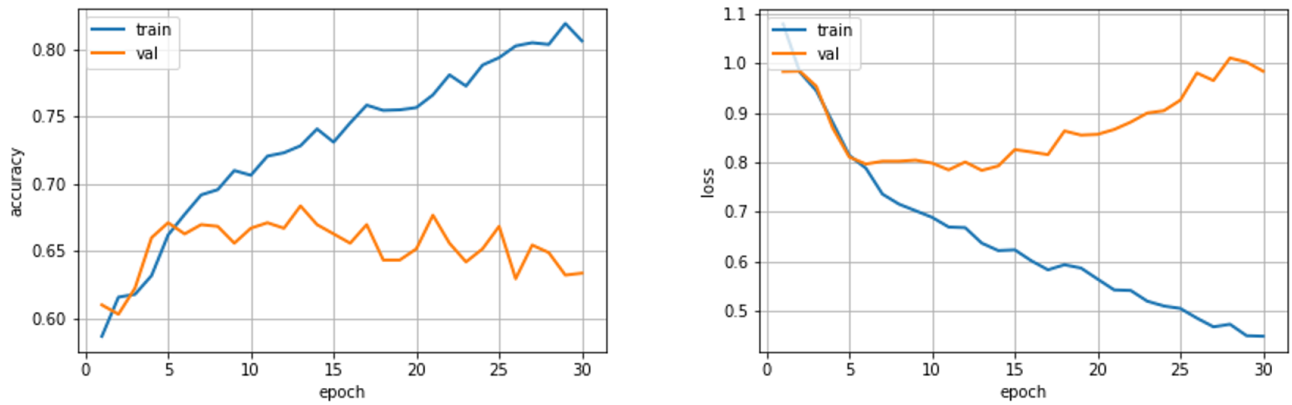

図 2　K=1 での BERT の訓練の様子．図左がエポックごとの訓練データ（train），検証データ（val）に対する精度（accuracy）の推移，図右がエポックごとの訓練データ（train），検証データ（val）に対する損失（loss）の推移である．

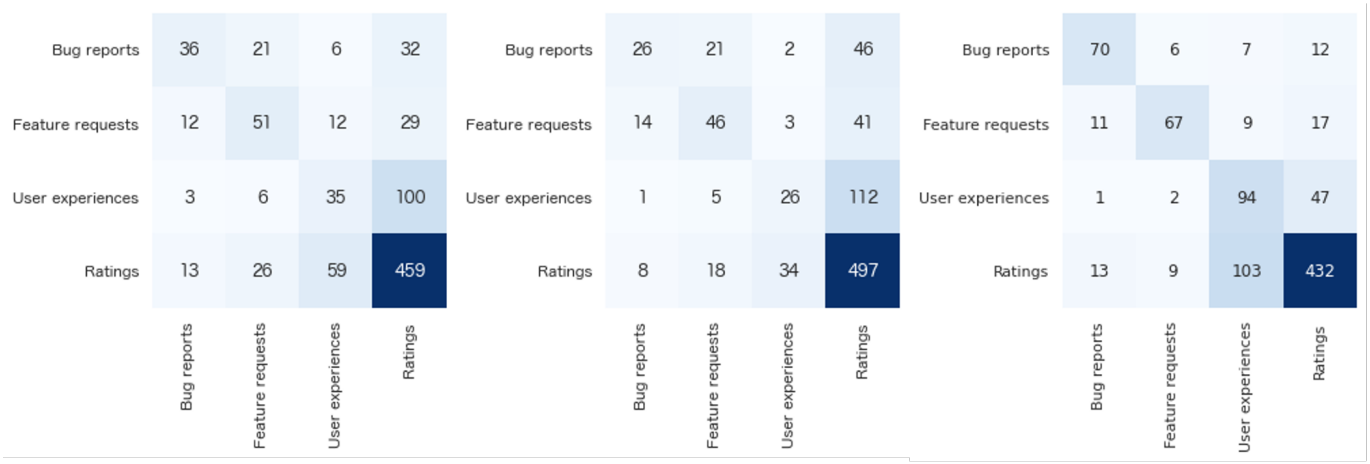

図 3　各分類器の混同行列．左から順に lemmatization + stopwords の条件で構築した Naive Bayes による分類器，stopwords の条件で構築したロジスティック回帰による分類器，BERT による分類器である．

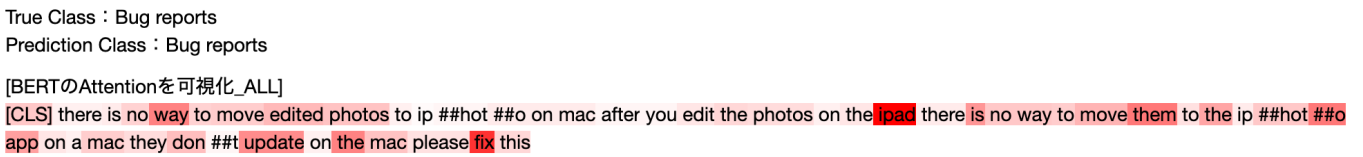

図 4　BERT の Attention の可視化

動でラベル付を実施した．ラベル付けは文献 [2] でのコーディングガイドにしたがって実施したが，ラベル付には主観が影響している可能性が考えられる．これについて追加で取得したアプリケーションレビューについて無作為に 50 件を抽出し，著者以外の 2 人にコーディングガイドに従いラベル付を依頼した．この 50 件のラベル付の一致度については Fleiss' kappa の係数によって評価した．この結果は $k = 0.5679$ であり，概ね一致していると言え，コーディングガイドにしたがうことでコーダー間のばらつきの影響は抑えられていると考えられる．一方，文献 [2] のデータセットと追加でラベル付を行ったデータセットではラベル付に差がある可能性が考えらる．また追加で取得したアプリケーションは特定のアプリに対するレビューだけであるため，その影響については今後は対象とするレビューデータを増やすことで影響を確認したい．

5.2　パラメータチューニングについて

次に構築した分類器のチューニングが性能に影響している点である．BERT でのパラメータチューニングの際は訓練データを 8:2 の比率で BERT 用の訓練データと検証データに分割して行った．よって BERT ではパラメータチューニングは行っているもののモデルの構築に使用しているデータ量は Naive Bayes およびロジスティック回帰による訓練時よりも 2 割少ない．一方で Naive Bayes およびロジスティック回帰のモデルではアルゴリズムのハイパーパラメータチューニングについては実施していないが特徴量エンジニアリングの段階で 4 つの実験条件を設定した．Naive Bayes およびロジスティック回帰ではアルゴリズムのパラメータチューニングを実施する

ことで性能が向上する可能性があるが，アルゴリズムのパラメータチューニングの影響で BERT の性能を越える可能性は低いと考えられる．この影響については今後，Naive Bayes およびロジスティック回帰にパラメータチューニングを加えた実験を実施することで確認したい．

6. むすび

本研究ではアプリケーションレビューを分類する機械学習モデルの構築と評価を実施した．機械学習モデルの構築アルゴリズムには Naive Bayes，ロジスティック回帰，BERT の 3 つを使用した．Naive Bayes では BoW による特徴量，ロジスティック回帰では fastText による特徴量，BERT では前処理を施した文章そのものを入力とした．なお Naive Bayes とロジスティック回帰では特徴量エンジニアリングの際に前処理を施した文章をそのまま用いる場合（default）と Stopwords を適用した場合（stopwords），レビュー文に含まれる各単語の見出し語化を適用した場合（lemmatization），Stopowrds と見出し語化の両方を適用した場合（lemmatization + stopwords）の 4 つの条件を設定した．3 つのアルゴリズムで構築した分類器の性能を比較した結果，BERT による分類器が最も高い性能 Precision 0.7237，Recall 0.7286，F1-Score 0.7173 を示した．なお Naive Bayes では lemmatization + stopwords の条件で構築した分類器が最も高い性能を示し，ロジスティック回帰では stopwords の条件で構築した分類器が最も高い性能を示した．

最も高い精度を示した BERT の分類器の混同行列から User experiences と Ratings のラベル間での誤判定が多く，2 つのラベルに含まれるレビュー文の類似性が示唆された．

今後はラベル間の類似性を考慮し単一のラベルを割り当てる多クラス分類器だけでなく，複数クラスを割り当てるマルチクラス分類器の構築を実施することも視野にいれる．また構築したモデルを実際に組み込み開発者にフィードバックを行うアプリケーションの開発なども考えられる．

謝辞

本研究で使用する追加のデータセットのラベル付に櫨山研究室の古川貴一さんに協力いただきました．心より感謝申し上げます．

文　献

[1] Stephan Krusche and Bernd Bruegge, “User Feedback in Mobile Development,” Proceedings of the 2nd International Workshop on Mobile Development Lifecycle MobileDeLi’14, pp. 25-26, 2014.

[2] Walid Maalej and Hadeer Nabil, “Bug Report, Feature Request, or Simply Praise? On Automatically Classifying App Reviews,” Proceedings of the IEEE 23rd International Requirements Engineering Conference, pp. 116-125, 2015.

[3] 山田 侑樹, 櫨山 淳雄, “fastText を用いたアプリケーションレビューの分類と局所的な説明の付与による考察,” ソフトウェアエンジニアリングシンポジウム, pp. 219-225, 2020.

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee and Kristina Toutanova, “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” arXiv preprint arXiv:1810.04805, 2018.

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin, “Attention Is All You Need,” arXiv arXiv:1706.03762, 2017.