No. 62008972

卒 業 論 文

論文題目

Google ストアと X における ユーザレビューを用いた 開発者支援ツール

指導教員

高 田 眞 吾

慶 應 義 塾 大 学 理 工 学 部

情 報 工 学 科

年 度

令 和 5 年 度

氏 名

佐 藤　響

論　文　要　旨

学 科	情報工学	学 籍 番 号	62008972	ふりがな 氏名	さとうひびき 佐藤響

(論文題名)

Google ストアと X における ユーザレビューを用いた開発者支援ツール

(内容の要旨)

モバイルアプリのレビューにはそのアプリに関するバグの報告や新しい機能の要望など開発者にとって有用な情報が多く存在する．しかし, レビューの数は膨大であり, 人の手で全て確認するのは不可能に近い．近年, トピック分類やキーフレーズ抽出などレビューをマイニングする技術が注目を集めているが, レビュー文は構造化されていないため分類の精度が上がりにくく, また, 細かい機能などの粒度の高い分類はできない．

本研究では, レビューの中にある問題のある機能やアプリに対する要望に関する報告を自動抽出してからその抽出した情報を元にクラスタリングすることによりクラスタリングの精度を向上させつつ, 粒度の高いクラスタリングを行う．また, その結果を時系列やアプリごとに Web ブラウザ上に出力し, 開発者を支援する可視化ツールを提案する．過去の論文で集められた同一期間に投稿された Google Play ストアのレビュー 7912 件と Twitter のツイート 1525211 件に加え, 本研究で新たに収集した Google Play ストアのレビュー 8000 件と Twitter のツイート 30000 件を機械学習を用いて, 有用な情報を記述している部分を抽出し, クラスタリング, web サイトにて可視化を行う．評価項目は以下の通りである．

- RQ1: レビューの抽出性能の調査
- RQ2: 抽出文を用いたクラスタリングの性能調査
- RQ3: 可視化ツールの有用性

調査の結果, レビューの自動抽出によりクラスタリング性能の向上と粒度の高さが確認された．また可視化ツールの有用性を示すことができた．

慶應義塾大学理工学部情報工学科高田研究室

Google ストアと X における
ユーザレビューを用いた開発者支援ツール

佐藤響

目次

図目次

表目次

第 1 章

序論

1.1 背景

1.1.1 Google Play ストア

Google Play ストア [12] とは, Google が提供している Android や ChromeOS 向けのデジタルコンテンツ配信サービスである. Google Play ストアを使用することにより, アプリやゲームの検索やインストール, 映画などのレンタルや漫画や書籍の購入などが可能となっている. Android 向けのアプリケーションストアであった “Android Market” が 2008 年から始まり, 2012 年 3 月 6 日に “Google Play ストア” と改名された. 2022 年 3 月をもって 10 周年となっており, 現在は数百万以上のコンテンツを配信している [1]. 2022 年 5 月時点で 190 カ国以上の 25 億人のユーザーが毎月 GooglePlay を使用しており, 収益は 1,200 億ドルに上る [3].

1.1.2 X

X(旧 Twitter)[13] とは, アメリカの X 社が運営している SNS サービスであり, 2023 年 7 月 24 日に前身の Twitter から名称を変更した. このサービスの主な機能は「フォロー」, 「ポスト (旧ツイート)」, 「リポスト (旧リツイート)」の 3 つである. 相手のユーザーをフォローすることにより相手の投稿を受け取ることができるようになる. ポスト (旧ツイート) とは, 自身の書き込みを投稿することであり, 2022 年時点では 1 日に 5 億件以上がポストされている [14]. また, 他人のポストをリポスト (旧リツイート) することにより自分のフォロワーに共有することができる. ポストする文章には全角で 140 文字, 半角で 280 文字の制限

があるものの, 有料の “Twitter Blue” に加入することによって, 全角で 2000 文字, 半角で 4000 文字までの文章をポストできるようになる.

1.1.3 アプリのユーザーレビュー

アプリのユーザレビュー (以下： レビュー) とはユーザがそのアプリをインストールして実際に使用した上で, そのアプリに対する評価やコメントをする機能のことである. このアプリのレビューにはユーザがアプリに対して抱いている不満やアプリへの賞賛が書かれる. また, そのアプリのバグの報告や新しい機能の要望などを記述されることもある. アプリの開発者はレビューを参考にしてバグの修正や新しい機能の追加などアプリの品質向上や保守に努めている.

1.2 本研究の目的

モバイルアプリのユーザレビューには, バグの報告や新しい機能の追加要望など開発者にとって有用な情報が多く存在する. しかし, そのレビューの数は膨大であり, 有用なレビューが埋もれてしまうことがある. したがって本研究では, GooglePlay ストアと X に存在するユーザレビューに含まれる有用な部分を自動で抽出し, その抽出結果をもとにクラスタリング, さらに web ブラウザ上で可視化することにより開発者をサポートする手法およびツールを提案する.

1.3 本論文の構成

本論文の構成を以下に示す.

第 2 章　関連知識・関連研究

本研究の関連研究であるアプリレビューのマイニング, ツイートとアプリレビューの関連性について述べる.

第 3 章　基盤技術

本研究で用いる基盤技術である機械学習, 自然言語処理, BERT, Chinese Whispers について述べる.

第 4 章　実装

本研究で提案しているスクレイピングから可視化までの一連の手法を説明する. 特にレビューに含まれる有用な箇所の自動抽出手法, 抽出結果をもとにクラスタリングする手法, web ブラウザにて出力する手法の 3 つの手法についてそれぞれ述べる.

第 5 章　結果・評価

本論文で実装した手法の評価実験とその結果について述べる. 3 つの Research Question をもとに結果を分析し, 考察する. 最後に妥当性の脅威について述べる.

第 6 章　結論

本研究の結論と研究を行う中で発見された今後の課題について述べる.

第 2 章

関連研究

2.1　アプリレビューのマイニング

近年, アプリレビューをマイニングするための自動化技術 (トピック分類やキーワード抽出など) に関する研究が進んでいる．これらの技術によって, 開発者がアプリレビューを理解・分析するために必要とする労力を軽減することに繋がっている．INFAR[6] はレビューから洞察を発見し, レビュー文を事前に定義されたトピックに分類したのちに要約を生成する手法である．定義されるトピックの粒度はクラッシュや GUI など粗いものとなっている．また, SUR-Miner[10] はレビューを表 2.1 に示した 5 つのカテゴリに分類し, 依存関係解析や Part-of-Speech パターンなどの技術を使用して, アプリレビューからいくつかの側面を抽出する．そして最後に概要を可視化する．この概要に関しては多くの開発者から有用性があると判断されている．他にも, Casper[11] というレビューからアプリの問題に関してユーザが報告したミニストーリー (ユーザーアクションと関連するアプリの動作という 2 種類のイベント) を抽出し, 合成するための手法が提案されている．

2.2　ツイートとアプリレビューの関連性

Gouri ら [4] は Twitter からのユーザフィードバックをタイミングと内容の 2 つの観点から評価し, App Store のレビューと比較した．ツイートとアプリレビューをテキスト分析して, LDA を用いて分類した．その結果, 426 件のツイートと 2,383 件のレビュー (バグレポートと機能リクエスト) のタイミング分析では, 約 15% が最初に Twitter に表示されることが示された．また, 15% のツイートのうち, 72% はモバイルアプリの機能または動作の側面に

表 2.1 Definition of Five Review Categories（[10] p.763, Table I）

Category	Definition	Examples
Praise	Expressing emotions without specific reasons	Excellent! I love it! Amazing!
Aspect Evaluation	Expressing opinions for specific aspects	The UI is convenient. I like the prediction text.
Bug Report	Reporting bugs, glitches or problems	It always force closes when I click the “.com” button.
Feature Request	Suggestions or new feature requests	It would be better if I could give opinion on it. It’s a pity it doesn’t support Chinese. I wish there was a “deny” button.
Others	Other categories that are defined in [31]	I’ve been playing it for three years

関連しているものであった. 一方で, App Store のレビューはモバイルアプリの機能または動作の側面に関連しているものが全体の 80% であった. さらに, 表 2.2 に示されているように, ツイートにはアプリに関連する重大な問題や深刻な問題を示すトピックがつぶやかれている事例を少なくとも 6 つ確認することができる. App Store のトピックもタイミングや回数などの詳細が追加されているものの, 同様の情報を示している.

表 2.2 Topic analysis from LDA（[4] p.20, Table IV）

App	Topic#	Topics on Twitter	Topics in App Store reviews
Dropbox	1	unable, file, sync, access, try	file, upload, unable, sync, horrible, time
	2	connect, fix, mac, open, crash	crash, every, time, try, three
Google cast	1	googlecast, work, bring, resolve, session	problem, fine, tv, connect, work
	2	reboot, router, tv, add, screen	googlecast, sometimes, screen, work, win
LinkedIn	1	wish, meet, announce, connect, use	option, add, thanks, vibrate, please
	2	use, prospects, connect, download, wish	say, make, account, launch, fight

第 3 章

基盤技術

3.1 機械学習

3.1.1 機械学習とは

機械学習とは, データを分析するための手法の 1 つであり, 大量のデータをコンピュータが学習し, データに潜んでいるルールやパターンといったものを発見する手法である. コンピュータが自ら学習した成果を用いて未知のデータの予測や発見を可能としている. コンピュータに反復的に学習させることでデータの中にある規則性, 特徴を発見することができる.

この技術は現在では, 生物学や自動運転, 金融工学などさまざまな分野で大きな影響を与えている.

機械学習の学習方法には教師あり学習, 教師なし学習, 強化学習の 3 種類が存在する.

3.1.2 教師あり学習

読み込んだデータから入力と出力の関係を学習させ, データ間の関係性を学習させる手法である. 学習データには事前に「正解」のラベルを付与される. 入力された値と正解のラベルのセットを繰り返し学習させることで, 未知の入力された値に対して正解となるデータを予測し, 出力することが可能となっている.

教師あり学習の具体例には需要予測や株価予測, 画像認識などが挙げられる.

3.1.3 教師なし学習

教師なし学習とは教師あり学習とは異なり, 学習データに「正解」のラベルは付与せずに, データセットのパターンからデータの関係を認知させる学習手法である. 正解と不正解が明確でない問題の解決策として用いられる. 与えられたデータを繰り返し学習することによりそのデータにどのようなパターンが存在するかをコンピュータ自身が見つけ出すことができ, 未知のデータに対する予測, 識別を可能とする.

教師なし学習で行う代表的な例は「クラスタリング」と「次元の削減」である. クラスタリングは複数のデータをそのデータの特徴に応じて幾つかのグループに分けることである. 次元の削減とはデータの次元数を減らすことでデータの特徴を表す情報を抽出することである.

3.1.4 強化学習

強化学習は, 環境と相互作用しながら報酬をもとに行動を学習する枠組みのことである. 強化学習には方策に従って行動を学習する主体である「エージェント」と状態と報酬をエージェントに返し, エージェントが行動を与える対象である「環境」の 2 つが存在する.

強化学習の具体例には将棋や囲碁などのゲーム AI やロボットの単純動作の獲得などが挙げられる. Google 社の SlphaGo という AI が韓国の囲碁プロ棋士に勝ったことで大きな話題を呼んだ学習方法である.

3.2 自然言語処理

自然言語処理 (Natural Language Processing) とは, 人間が使用する言語 (自然言語) をコンピュータが分析する技術である. 自然言語処理によって大量の自然言語によるテキストデータが持つ意味を解析, 処理することができる. 自然言語には文脈によって解釈が変わるなどの言葉の曖昧性や意味の重複といったものが含まれている. 従ってコンピュータがそれを処理するのは難しいため課題とされている.

自然言語処理の活用事例として質問への回答, 文の要約や翻訳, テキスト分類などが挙げられる.

3.3 BERT

3.3.1 BERT とは

BERT(Bidirectional Encoder Representations from Transformers)[5] とは, Google の Jacob Devlin らによって 2018 年秋に提案された言語表現モデルである. BERT は, ラベル付されていないテキストから深い双方向表現を事前学習するように設計されている. その結果, 質問応答や言語推論などの幅広いタスクのためのモデルを作成するために, 出力層を 1 つ追加するだけで微調整することができる. BERT は 11 の自然言語処理タスクにおいて, GLUE スコアや SQuAD v1.1 による質問応答テストの F1 スコアなどで向上が確認された [5].

3.3.2 学習方法

BERT は事前学習とファインチューニングの 2 つのステップからなり, どちらのステップにおいても transformer モデルを用いる. transformer とは深層学習のベースとなっているモデルである. RNN や CNN の並列処理ができないという欠点がある中で, transformer は再帰や畳み込みは一切行わず, Attention のみを用いることで並列化を可能にした. 基本的な構成は multi-head attention 層, add&nom 層 (残差結合&layer normalization), position-wise FNN 層となっている.

3.3.3 事前学習

事前学習では, Masked Language Model と Next Sentence Prediction の 2 つのタスクを解く. Masked Language Model は入力されたトークンをランダムにマスクし, マスクされたトークンを他のトークンから予測するタスクである. Next Sentence Prediction ではある文章に対して, その後に出現する文を並べたペアを正例, ランダムな文章を並べたペアを負例として識別する問題を解く. この問題を解くことにより, 2 つの文章が隣り合っているかどうかを予測するよう学習する. この 2 つのタスクは自己教師あり学習である. 自己教師あり学習とは, ラベルが付与されていない大量のデータセットを用いて, プレテキストタスク (擬似的なラベルが自動生成された代わりのタスク) を解くための事前学習を行うための学習方法である. 自己教師あり学習により人の手作業によるラベル付を必要とせずに大量のデータで学習することが可能である.

3.3.4 ファインチューニング

ファインチューニングでは, ラベル付きデータを用いて特定のタスクに特化するように学習させる. 3.1.1 で述べた通り, 解きたいタスクに応じて transformer の上に出力層を 1 つ追加する. そしてラベル付きデータを用いて出力層と transformer のパラメータを更新する.

3.4 Chinese Whispers

3.4.1 Chinese Whispers とは

Chinese Whispers(CW)[2] とはグラフクラスタリングのランダム化アルゴリズムである. CW は, 事前にクラスタの数を指定せずに, クラスタの数を自ら選択することができるため, 異なるサイズの分類を扱うことができる. そのため, クラスタの数が事前にわからない NLP 問題に適している.

3.4.2 グラフの構築

まず, ノードとエッジからなる, 重み付き無向グラフを作成する. これがクラスタリングの対象となるグラフである. ノードはテキスト文書などのデータの要素を表し, エッジはノードの関連性を表す.

3.4.3 クラスタリングの開始

クラスタリングの開始時点では, 各ノードはそれぞれ異なるクラスタに属するものとする. すなわち, 1 つのノードに対して 1 つのクラスタが割り振られる.

3.4.4 グラフの反復処理とクラスタの更新

ノードは少数の反復ステップによってグラフを処理し, 接続されたノードの情報を受け取り, 接続するノードの中で最も近いノードのクラスを継承する. これは現在のノードに対するエッジの重みが最大となるクラスタであるため, 類似したノードが同じクラスタにグループ化される. 最も強いクラスが複数ある場合はランダムで 1 つ選ばれる. 一方でどのエッジにも接続されていないノードはクラスタリングのプロセスから除外されるため, 一部のノー

ドはクラスタリングされないことがある.

3.4.5 クラスタリングの収束

このグラフの反復処理とクラスタの更新を繰り返す．これによりクラスタが再構築され, 新たなクラスタの構築が進行する．これをクラスタリングが収束するまで繰り返す．この結果, 各ノードは最終的なクラスタに所属する.

第 4 章

実装

4.1 実装の概要

本研究では, 膨大の数ある GooglePlay と X に投稿されたアプリレビューをスクレイピングして取得し, レビュー文に含まれるバグレポートやアプリに対する要望を示す箇所を自動抽出, その結果を利用しクラスタリング, 最後に web ブラウザ上に可視化する手法を提案する.

実装環境

- オペレーティングシステム
  - Mac OS Ventura 13.4.1
- 実装言語
  - Python 3.11.6

4.2 対象アプリとレビュー

本研究では川面による先行研究 [17] のデータセットを使用するため対象アプリは先行研究のアプリに合わせている. 今回使用するレビューのデータセットは先行研究によって作成された 13 個のアプリレビューのデータセットに加え, 本研究で新たに収集したアプリレビューのデータセットを使用する. 対象アプリを先行研究に合わせた理由としては, 現在, X のポスト取得数に制限がある影響で十分な数のツイートが用意できなかったことが原因である. X のポスト取得数の制限に関しては 4.4 で詳しく述べる. 対象となっているアプリを表

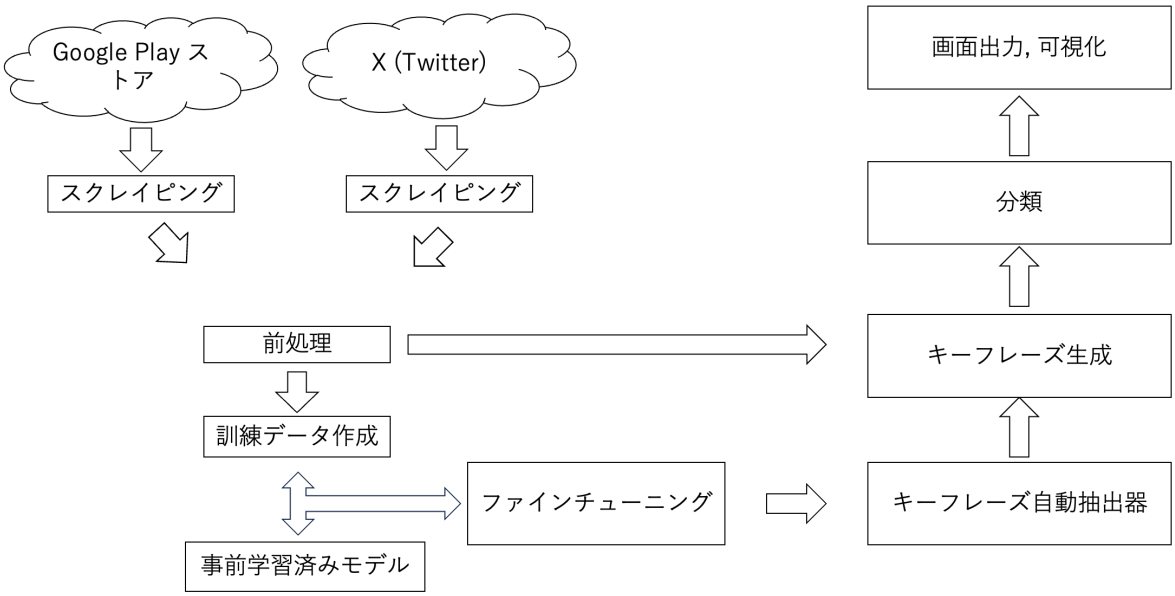

図 4.1　実装した提案手法の流れ

4.1 に示す.

4.3　Google Play ストアのスクレイピング

本研究で取得するレビュー情報は先行研究に合わせて下記とする.

- reviewId：レビュー ID
- userName：ユーザ名
- userImage：ユーザのプロフィール画像
- at：投稿日時
- score：星の数
- content：レビュー内容
- thumbsUpCount：このレビューが参考になったと評価した人の数
- reviewCreatedVersion：レビュー時のバージョン
- replyContent：開発者からの返信の内容

表 4.1　本研究の対象アプリ一覧

アプリ名(一部略称)	Google Play ストアのパッケージ ID	Twitter の 検索キーワード
にゃんトーク	com.akvelon.meowtalk	にゃんトーク
スマートニュース	jp.gocro.smartnews.android	スマートニュース
PayPay	jp.ne.paypay.android.app	paypay
Coke ON	com.coke.cokeon	coke on
Google Fit	com.google.android.apps.fitness	google fit
Simeji	com.adamrocker.android.input.simeji	simeji
Lemon8	com.bd.nproject	lemon8
楽天ペイ	jp.co.rakuten.pay	楽天ペイ
majica	com.donki.majica	majica
LINE MUSIC	jp.linecorp.linemusic.android	line music
BuzzVideo	com.ss.android.article.topbuzzvideo	buzzvideo
ファミペイ	jp.co.family.familymart_app	ファミペイ
CapCut	com.lemon.lvoverseas	capcut

- repliedAt：開発者からの返信日時

先行研究では投稿日時が 2021 年 10 月 21 日〜2021 年 12 月 15 日までの 8 週間のレビューを収集している. 用意された Google Play ストアの各アプリのレビュー数は表 4.2 の通りである. この先行研究のデータに加え, 本研究では 2023 年 10 月 1 日 12 月 15 日のレビューを新たに取得する. 新たに取得された Google Play ストアの各アプリのレビュー数は表 4.3 の通りである.

Google Play ストアのレビューをスクレイピングするために Python のプログラムである (`get_google_play_review.py`) を作成した. このプログラムの作成にあたり, Python のライブラリである google-play-scraper を使用する. google-play-scraper では外部依存関係なしで Python 用の Google Play ストアを簡単にクロールするための API が提供されている [9]. このライブラリを使用することにより, アプリのパッケージ名, 言語, 取得する数, 順

表 4.2　収集した Google Play ストアのレビュー数（[17] p.16, 表 4.2）

アプリ名	収集したレビュー数（件）
にゃんトーク	171
スマートニュース	1,651
PayPay	1,052
Coke ON	1,736
Google Fit	372
Simeji	468
Lemon8	72
楽天ペイ	480
majica	706
LINE MUSIC	359
BuzzVideo	375
ファミマのアプリ	290
CapCut	180
合計	7,912

序を指定してレビューの一覧を取得することができる.

4.4　X のスクレイピング

本研究で取得するツイート情報は先行研究に合わせて下記とする.

- id：ツイート ID
- content：ツイート内容
- at：ツイート日時

まず, 先行研究で収集したツイート数を表 4.4 に示す.

この先行研究に加え, 本研究では新たに 2023 年 10 月 1 日 12 月 15 日のポストを取得する．X のポスト取得に関しては Twitter API を使用してスクレイピングを行う．Twitter

表 4.3　収集した Google Play ストアのレビュー数 (2023/10/1〜12/15)

アプリ名	収集したレビュー数（件）
にゃんトーク	
スマートニュース	
PayPay	
Coke ON	
Google Fit	
Simeji	
Lemon8	
楽天ペイ	
majica	
LINE MUSIC	
BuzzVideo	
ファミマのアプリ	
CapCut	
合計	

API のプランに関しては Free, Basic, Pro, Enterprise の 4 つのプランが用意されておりそれぞれ料金や使用できる機能などが異なる．大規模なサービスやビジネス向けの Enterprise プラン以外の 3 つのプランの違いの一部を表 4.5 に示す．表 4.5 よりポストを取得するためには Basic プラン以上に加入する必要がある．Basic プランに加入した場合でも合計で 30,000 件しか取得されないため本研究では先行研究のデータセットを追加で使用することとした．X の利用規約によると, X が提供するインターフェイスを介して行うスクレイピング以外は禁止としている．そのため, selenium などを使用したスクレイピングは断念し, 過去の論文のデータを使用することとした．

Twitter API を使用してポストを取得するために Python のプログラムである (`get_tweet.py`) を作成した．このプログラムでは Twitter API にアクセスするためのライブラリである Tweepy[15] を使用した．まず API キーなどの 4 つの認証情報をセットする．次に Client クラスの search_resent_tweet メソッドを使用してツイートを取得する．こ

表 4.4　収集した Twitter のツイート数（[17] p.18, 表 4.3）

アプリ名	収集したツイート数（件）
にゃんトーク	2,525
スマートニュース	50,590
PayPay	880,319
Coke ON	84,424
Google Fit	13,496
Simeji	205,327
Lemon8	4,376
楽天ペイ	11,111
majica	3,649
LINE MUSIC	184,873
BuzzVideo	41,656
ファミマのアプリ	8,867
CapCut	33,998
合計	1,525,211

表 4.5　プランとできること

	Free	Basic	Pro
料金	無料	月額 100 ドル	月額 5,000 ドル
月間ポスト数の上限	1,500	3,000	300,000
月間ポスト取得数	0	10,000	1,000,000

のメソッドは最大過去 7 日間まで遡ってツイートを取得できる. search_all_tweets メソッドでは全てのツイートを取得できるが, “Academic Research” という学術用の用途で API 承認されたユーザーしか使用できないため本研究では使用しなかった. 先行研究と同じ情報を取得するために, 本研究では引数として以下のものを与えた.

- max_resul : 検索結果の最大数. 10〜100 の数値で, デフォルトは 10
- query: 検索ワード
- tweet_field: ツイートフィールドを選択. 今回はツイート日時を取得するために [”created_at”] とした.
- end_time: 期間の終わりを指定できる (UTC タイムスタンプ)

新たに取得した X のポスト数を表 4.6 に示す

表 4.6 収集した X のポスト数

アプリ名	収集したツイート数（件）
にゃんトーク	
スマートニュース	
PayPay	
Coke ON	
Google Fit	
Simeji	
Lemon8	
楽天ペイ	
majica	
LINE MUSIC	
BuzzVideo	
ファミマのアプリ	
CapCut	
合計	

4.5 前処理

機械学習によるレビューに含まれる有用な箇所の自動抽出の精度を上げるために, GooglePlay ストアと X から取得したデータに対して前処理を行うプログラム

(preprocessing_google.py, preprocessing_twitter.py) を作成した．この処理では以下に示す処理を行う．この処理は一般的な自然言語処理の手法を参考としている．

- 英語を全て小文字に揃える．
- 以下の文字列を削除.
  - 「」【】（）()『』
  - @@から始まるメンション
  - #から始まるタグ
  - URL
  - 半角空白，全角空白
  - 絵文字
  - 日本語を含まないレビュー
- レビューやツイートには, 異なるバグの報告や新しい機能の要望に関する文が 2 文以上からなるものがある．そのため，「。」「.」「！」「!」「？」「!」「\n」「\r\n」でそれぞれの文に分割する．

以下の図 4.2 がレビューを前処理した例である．2 つの文で構成されているため「。」で区切り分割している．また絵文字は削除されている．

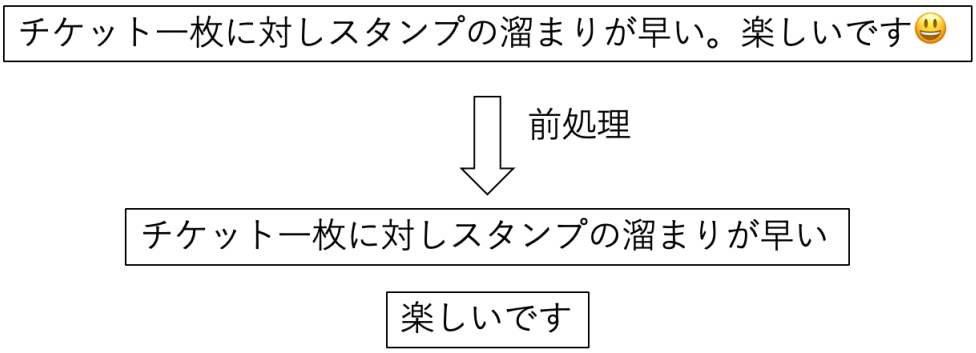

図 4.2　前処理の例

前処理した結果を csv ファイルにて保存する．保存する項目としては, 投稿日時 (at), レビューの id(reviewId) またはツイート id(id), そして, 前処理した文章である．図 4.7, 図 4.8 に前処理結果後の csv ファイルの一部を示す．

表 4.7　Google Play ストアレビューの前処理結果 (buzzvideo)

at	reviewId	content
2021-12-15 19:25:30	gp:AOqpTOHj6w ...	バズビデオを見て、感動をありがとう
2021-12-15 12:28:09	gp:AOqpTOHleV ...	内容が残酷で異常な人が多い
2021-12-15 11:09:50	gp:AOqpTOHG7O ...	分かりづらい
2021-12-14 15:16:33	gp:AOqpTOGWvT ...	ばず 29 さいって人が投稿してる動画すべて虚偽動画なのでアカウント削除と動画削除して欲しい
2021-12-14 15:16:33	gp:AOqpTOGWvT ...	あるだけで大迷惑です
2021-12-14 15:16:33	gp:AOqpTOGWvT ...	二度と登録し直せないよう個体識別番号で縛ってください
2021-12-14 15:16:33	gp:AOqpTOGWvT ...	お願いします

表 4.8　ツイートの前処理結果 (BuzzVideo)

at	id	content
2021-12-15T23:55:11.000Z	1471267626655825922	芸能人に似てる気がするけど名前が思い出せない
2021-12-15T23:53:43.000Z	1471267256659509249	驚愕男性が豆乳を飲むべき 3 つの理由
2021-12-15T23:53:43.000Z	1471267256659509249	男だからこそ注目したい豆乳のメリットとは
2021-12-15T23:53:17.000Z	1471267149746679813	感情を乗せた歌声と歌詞に聞き惚れちゃう♪壊れかけの radio
2021-12-15T23:53:10.000Z	1471267120264904705	kk と眞子の酷い嘘
2021-12-15T23:53:10.000Z	1471267120264904705	恐ろしい真実が明らかに

4.6 有用な箇所の自動抽出

4.6.1 概要

膨大な数あるレビュー文から開発に有用なレビューを絞り込み, かつ有用なレビュー文の中からバグの報告やアプリに対する要望に関して記述している部分を自動抽出する. 自動抽出のために日本語のデータで事前学習済みの言語表現モデルである日本語 BERT に対して質問応答形式の fune-tuning を行うことで自動抽出器を生成する. 抽出を行う文章の特徴をモデルに理解させるために質問文にその文章が GooglePlay ストアのレビューなのか X の文章なのかという「カテゴリー」の情報と「アプリ名」という 2 つの情報を加える. 図 4.3 に質問応答形式による fine-tuning のモデルを示す.

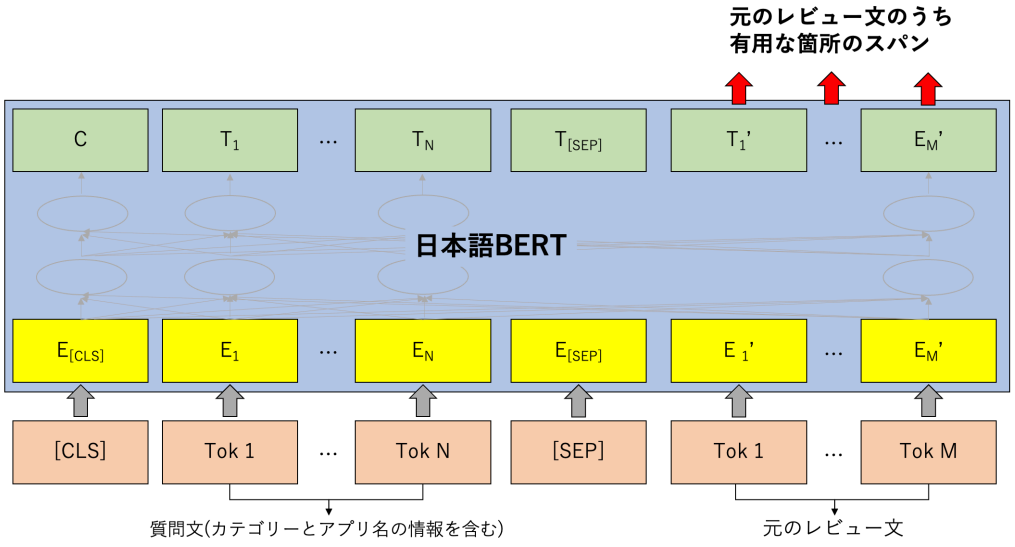

図 4.3 ファインチューニング

また, 図 4.4 に質問文とその答えの例を示す. 質問文にアプリの欠陥やアプリに対する要望を尋ねる文章を与え, その答えとして欠陥や要望を示す箇所を返すようにしている.

4.6.2 データセット

データセットとして, Google Play ストアと X のツイートからそれぞれ 5,000 件ずつ合計で 10,000 件のデータをランダムに抽出し手作業で有用な箇所を抽出した. データセット

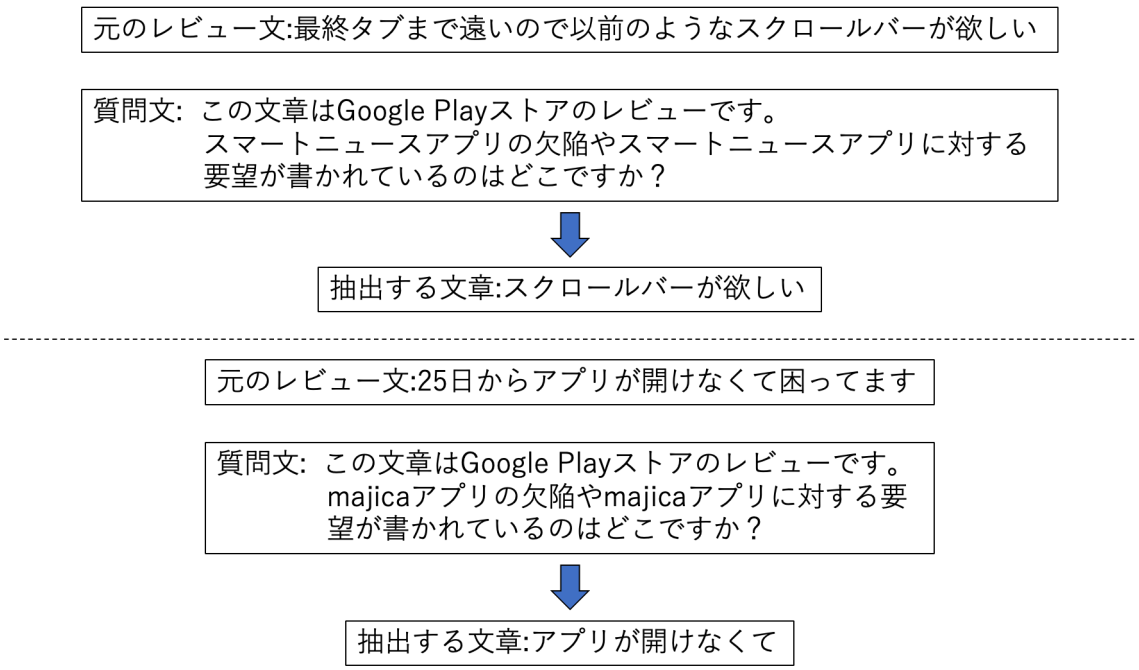

図 4.4　有用な箇所の抽出例

の作成は情報工学科の学部 4 年生 2 人がそれぞれ手作業で行い, お互いの抽出した箇所が異なっていたものは議論することにより決定した. データセットには前処理したデータを利用した csv ファイルを使用する. この csv ファイルには id, アプリ名, 投稿日時, 本文, 手動で抽出した結果の 5 つの情報が入っている. id は前処理したデータを識別するために与えられ, Google Play ストアのレビュー文は g_index, twitter のツイートの id は t_index とする. 10,000 件のデータセットのうち, 6,000 件を訓練データ, 2,000 件を検証用データ, 2,000 件をテストデータとする. 質問応答形式の fine-tuning を行うために, csv 形式であるデータセットをソースコード 4.1 に示すように json 形式に変換する.

ソースコード 4.1　データセット.json

```
1:  {
2:    "version": "v2.0",
3:    "data": [
4:      {
5:        "title": "モバイルアプリのレビュー",
6:        "paragraphs": [
7:          {
8:            "context": "本アカウントのフォローやリツイートお願いします",
9:            "qas": [
10:             {
11:               "id": "t_2223388",
```

```
12:                 "question": "この文はTwitterのツイートです。
13:                              paypayアプリの欠陥やpaypayアプリに対する
14:                              要望が書かれているのはどこですか？",
15:                 "is_impossible": true,
16:                 "plausible_answers": [{"text": "", "answer_start": -1}],
17:                 "answers": [{"text": "", "answer_start": -1}]
18:               }
19:             ]
20:           },
21:           {
22:             "context": "11/25前後からアプリを開いても強制終了、
23:                        会員バーコードもクーポンも何も出せない状態、
24:                        これでは買い物ができないと、こちらのレビュー
25:                        を見に来て沢山の方が同じ状態であることが
26:                        わかった",
27:             "qas": [
28:               {
29:                 "id": "g_6041",
30:                 "question": "この文章はGooglePlayストアのレビューです。
31:                              majicaアプリの欠陥やmajicaアプリに対する
32:                              要望が書かれているのはどこですか？",
33:                 "answers": [{"text": "アプリを開いても強制終了、
34:                                       会員バーコードもクーポン
35:                                       も何も出せない",
36:                              "answer_start": -1}],
37:                 "is_impossible": false
38:               }
39:             ]
40:           }, ...
41:         ]
42:       }
43:     ]
44:   }
```

この json ファイルは下記の構成になっている.

- version: バージョンを表す. 今回は答えられない質問を含む SQuAD 2.0 と同じバージョンのため, v2.0 とする
- title: context のタイトル
- paragraphs: context1 つとそれに関連する質問, 答えがリスト形式で保持されている
- qas: 質問と回答がリスト形式となっている
- context: 元の文章 (抽出する前の文章)
- id: 設定した id
- question: 質問文
- is_impossible: 答えられない質問なら true, それ以外は false
- plausible_answers: 質問が答えられない時のみ存在し, 問題文から答えになりうる部分を抽出
- answers: context から抜き出した答えとその位置情報がリスト形式で保持されてい

る. 答えを複数用意することもできる.

- text: context から抜き出した答えのテキスト情報 (抽出する文章)
- answe_start: context から抜き出した答えの位置情報

4.6.3 モデルの fine-tuning

用意したデータセットを用いてモデルを fine-tuning する. 本研究では事前学習済みモデルを提供するフレームワークである Huging Face の Transformers を通して利用できる東北大学のモデル [8] を使用する. このモデルは日本語の Wikipedia のデータを用いて学習されている [8]. この東北大学が公開している日本語 BERT のうち, whole word masking を適用して学習させているモデル [7] を用いる. whole word masking とは事前学習時に単語ごとでマスクするかどうかを決め, マスクする単語に対応するサブワードを全てマスクする方式である. モデルのパラメータは以下に示す通りである.

- 学習率: 3e-5
- エポック数: 10
- バッチサイズ: 12

実装には Transformers に含まれるスクリプトである run_squad.py を用いる.

4.6.4 自動抽出

fine-tuning を行ったモデルを使用して自動抽出を行う. 前処理した GooglePlay ストアのデータ 14,051 件と, twitter のデータ 4,634,319 件のデータから有用な箇所を自動抽出する. これにより開発に有用なレビューのみを選択でき, その文章の中に含まれるバグの報告やアプリに対する要望に関して記述している部分を抽出できる.

結果は表 4.9 に示すように csv 形式で保存する.

4.7 クラスタリング

抽出した文章のクラスタリングには既存研究のクラスタリング手法 [16] を参考にして実行する. この手法は以下の 3 つの手順である.

表 4.9　Google Play ストアレビューの自動抽出結果 (google_fit)

id	app_name	datetime	context	prediction
g_955	coke_on	2021-11-27 11:17:03	商品が出ない事が何回か発生しました	商品が出ない
g_956	coke_on	2021-11-02 12:15:37	使用している端末が、利用できる端末の一覧表にないため、サポートは期待できない	使用している端末が、利用できる端末の一覧表にない
g_959	coke_on	2021-11-11 15:32:50	そもそも自販機側が黄色点滅していなくて買えないことが多過ぎです	自販機側が黄色点滅していなくて買えない
g_961	coke_on	2021-11-14 23:13:26	今までは coke_on 対応を優先してかっていたが、これからはコカコーラ製品全般をできるだけ買わないようにする	今までは coke_on 対応を優先してかっていたが、これからはコカコーラ製品全般をできるだけ買わないようにする
g_964	coke_on	2021-10-24 12:30:07	自販機との接続を早くしてほしい	自販機との接続を早くしてほしい
g_965	coke_on	2021-11-11 16:43:39	やっと繋がっても先にキャンペーン広告が出てすぐに買えないのが不親切	キャンペーン広告が出てすぐに買えない
g_969	coke_on	2021-12-07 08:28:27	コークオンパスのフリー 20 プランの残り回数が分かりやすく表示してほしい	コークオンパスのフリー 20 プランの残り回数が分かりやすく表示してほしい
g_973	coke_on	2021-11-23 14:51:45	反応しない	反応しない
g_977	coke_on	2021-11-11 18:22:04	2 本以上の購入はとても使えません	2 本以上の購入はとても使えません
g_978	coke_on	2021-11-01 11:06:09	それと対応自販機との連携が悪い	自販機との連携が悪い
g_980	coke_on	2021-10-22 03:22:06	動きが遅い	動きが遅い
g_982	coke_on	2021-11-22 11:58:45	ただ自分のスマホのストレージが小さく、データ容量が大きいため今回一先ず削除いたします	自分のスマホのストレージが小さく、データ容量が大きい

1. Universal Sentence Encoder (USE) を用いて, 問題のある特徴語句を 512 次元のベクトルに変換
2. 重み付き無向グラフを構成し, 各問題機能をノードとし, 2 つの問題機能の USE ベクトル間のコサイン類似度スコア (比率) をノード間の重みとする. 比率は入力ハイパーパラメータであり, 問題のある機能間の意味的相関を測定. ハイパーチューニングした閾値は 0.5 とした
3. このグラフに対して, Chinese Whispers (CW) を実行し, 問題のある特徴量をクラスタリング

既存研究では英語の文章をベクトルに変換するために Universal Sentence Encoder (USE) を使用しているが, 今回は対象が日本語の文章であるため, Sentence-BERT の日本語モデルを使用する. Sentence-BERT とは, 事前学習された BERT モデルと Siamese Network を使い, 高品質な文ベクトルを作る手法である. このモデルを使用することで, 高品質な文ベクトルが作成できる. したがって本研究でのクラスタリング手法は下記である.

1. Sentence-BERT の日本語モデルを用いて, 抽出した文章をベクトルに変換
2. 重み付き無向グラフを構成し, 各問題機能をノードとし, 2 つの抽出した文章間のベクトル間のコサイン類似度スコア (比率) をノード間の重みとする. 比率は入力ハイパーパラメータであり, 抽出した文章間の意味的相関を測定. 検証した結果, 閾値は 0.8 とした
3. このグラフに対して, Chinese Whispers (CW) を実行し, 問題のある特徴量をクラスタリング

これにより, 抽出した文章をその文章が示す意味に応じてクラスタリングすることができる. それぞれの文章にクラスタの番号 (以下： クラスタ番号) が振られ, クラスタ番号が同じものが同じクラスタとなり番号が近いものは意味的相関が近いことを表す.

結果は csv ファイルに保存される. 表 4.10 に示されるように抽出した文章にクラスタ番号が振られる.

4.8 画面出力・可視化

クラスタリングした結果を web ブラウザ上で可視化する. web アプリケーションの実装に使用した言語, フレームワークは以下となっている.

表 4.10　抽出した文章とクラスタ番号 (google_fit)

prediction	cluster
十分歩いて 108 歩とかふざけんな	273
278 歩に減っていた	274
再起動しても直らない	275
接続/連携を適宜確認しておく必要がある	276
歩いた歩数より足りない	279
使えない	280
使えない	280
歩けない	280
使えない	280
動かなかった	280
反応しない	280
使い方も分からない	280
使えない	280
使えない	280
動かなくなった	280
長期放置されてるんでしょうか	281
下がるって何故でしょうか	282
カウントされなくなる	283
何もカウントしなくなった	283
カウント出来ていない	283
カウントされず	283
記録ができませんと	283
計測しなくなった	283
カウントされなくなった	283
カウントしない	283
全くカウントされていない	283
カウントしなくなりました	283
データが反映されなくなった	283

- フロントエンド: HTML/CSS, JavaScript
- バックエンド: Python
- フレームワーク: Flask

アコーディオンメニューを使用して抽出した文章の一覧をクラスタごとに表示する．また, 抽出した文章をクリックすると投稿日時や元のレビュー文がモーダルウィンドウで表示する．さらにレビューが投稿された期間やレビュー文のキーワードで絞り込み検索することが可能となっている．そして, 日ごとのレビュー数を表す折れ線グラフとクラスタに含まれるレビュー数の上位 10 個を表す棒グラフを作成した．この 2 つのグラフは検索結果に応じて動的に変化するようになっている．

第 5 章

結果・評価

5.1　RQ1: レビューの抽出性能の調査

自動抽出器の抽出性能について調査する.

5.2　RQ2: 抽出文を用いたクラスタリングの性能調査

抽出された文章のクラスタリングの性能について調査する.

5.3　RQ3: 可視化ツールの有用性

抽出, クラスタリングによって得られた結果を表示する可視化ツールについて調査する.

第 6 章

結論

レビューの中にある問題のある機能やアプリに対する要望に関する報告を自動抽出してからその抽出した情報を元にクラスタリングすることによりクラスタリングの精度を向上させつつ, 粒度の高いクラスタリングを行う. また, その結果を時系列やアプリごとに Web ブラウザ上に出力し, 開発者を支援する可視化ツールを提案した.

3 つの RQ にて自動抽出, クラスタリングの性能を評価し, 可視化の有用性を示すことができた.

謝辞

本研究を行うにあたり, 日頃から親身にご指導をしていただいた慶應義塾大学理工学部情報工学科の高田眞吾教授に感謝いたします. そして, 未熟な私を様々な面で身近で支えていただいた慶應義塾大学理工学部情報工学科 高田研究室の先輩方, 同期の皆様に心から感謝いたします.

2024 年 1 月

佐藤　響

参考文献

[1] “android の「google play ストア」では何ができる？ アプリのインストール方法や楽しみ方を知ろう”. https://www.android.com/intl/ja_jp/articles/120/, 2023. Accessed: Nov 8, 2023.

[2] Chris Biemann. Chinese whispers: An efficient graph clustering algorithm and its application to natural language processing problems. In *Proceedings of the Association for Computational Linguistics*, 2006.

[3] Google Developers Blog. Celebrating 10 years of google play. together. https://android-developers.googleblog.com/2022/03/celebrating-10-years-of-google-play.html, mar 2022. Accessed: Nov 8, 2023.

[4] Gouri Deshpande and Jon Rokne. User feedback from tweets vs app store reviews: An exploratory study of frequency, timing and content. In *5th International Workshop on Artificial Intelligence for Requirements Engineering*, pp. 15–21, 2018.

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv*, pp. 4171–4186, 2019(2018).

[6] Cuiyun Gao, Jichuan Zeng, David Lo, Chin-Yew Lin, Michael R. Lyu, and Irwin King. Infar: Insight extraction from app reviews. In *Proceedings of the 2018 ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, pp. 904–907, Lake Buena Vista, FL, USA, November 2018. ACM.

[7] GitHub. cl-tohoku/bert-base-japanese-whole-word-masking. https://huggingface.co/cl-tohoku/bert-base-japanese-whole-word-masking.

[8] GitHub. cl-tohoku/bert-japanese. https://github.com/cl-tohoku/

bert-japanese.

[9] Google Play Scraper Contributors. google-play-scraper, 2023. Accessed: Nov 10, 2023.

[10] Xiaodong Gu and Sunghun Kim. What parts of your apps are loved by users? In *30th IEEE/ACM International Conference on Automated Software Engineering, ASE 2015*, pp. 760–770, Lincoln, NE, USA, November 9-13 2015.

[11] Hui Guo and Munindar P. Singh. Caspar: Extracting and synthesizing user stories of problems from app reviews. In *Proceedings of the 42nd International Conference on Software Engineering (ICSE '20)*, pp. 628–640, June 27 - July 19 2020.

[12] Google Inc. "google play". http://play.google.com/.

[13] Twitter Inc. "twitter". https://twitter.com/.

[14] Time. Why elon musk wanted twitter in numbers. https://time.com/6171871/why-elon-musk-wanted-twitter-in-numbers/, 2022. Accessed: Nov 8, 2023.

[15] Tweepy Developers. Tweepy. https://www.tweepy.org/, 2023.

[16] Yawen Wang, Junjie Wang, Hongyu Zhang, Xuran Ming, Lin Shi, and Qing Wang. Where is your app frustrating users? In *44th International Conference on Software Engineering*, pp. 2427–2439, 2022.

[17] Kawatsura Yoshiki. モバイルアプリケーションストアと twitter におけるユーザレビューの比較調査. 2022(2021).