Vol.2018-DPS-175 No.9
Vol.2018-MBL-87 No.9
Vol.2018-ITS-73 No.9
2018/5/24

深層学習を用いた位置依存性の高いモバイルアプリ抽出手法

落合 桂一[1,a)]　ファティナ プテリ[1]　深澤 佑介[1]

概要：本研究では，場所に関連したモバイルアプリ（ローカルアプリ）を抽出する手法を提案する．アプリストアで開発者がアプリを登録するときに予め定められたカテゴリを登録するが，ローカルアプリを指定するカテゴリは存在しないため，アプリの内容から判定する必要がある．本研究では，アプリの名称や紹介文から生成した特徴量に基づき，ローカルアプリに該当するか否かを 2 値分類する．Google Play のデータを対象に評価を行い，提案手法の有効性を確認した．

1. はじめに

スマートフォンの普及により，スマートフォン上で利用するモバイルアプリが日常生活でよく利用されている．モバイルアプリの中には，特定の地域の観光向けアプリや，特定地域の居住者向けに自治体が提供するアプリなど，実世界に関する情報を提供するモバイルアプリが存在する．本研究ではそのようなモバイルアプリをローカルアプリと呼ぶ．ローカルアプリではユーザの行動を支援する情報を提供しており，スマートフォンユーザにとって有用である．

Google Play[*1] や App Store[*2] などのモバイルアプリ提供プラットフォーム（以下，アプリストア）では，非常に多くのアプリが提供されている．アプリストアでは予め定められたカテゴリがあり，各アプリはアプリ提供者によってカテゴリが付与されている．しかしながら，ローカルアプリを指定するカテゴリは存在しないため，ユーザは様々なキーワードで検索を行いローカルアプリを検索する必要がある．ローカルアプリを分類できれば，ある地域で有用なアプリをユーザの位置情報に基づいて推薦したり，アプリ紹介サイトで地域ごとのアプリを紹介するなどの応用が考えられる．

モバイルアプリの分類に関する従来研究では，アプリ名やアプリ名で Web 検索を行い取得したスニペットを文書と見なし，それらの文書に出現する単語を特徴量として短文（short text）の分類問題としてアプリの分類が定式化されている [1], [2]．一般に，短文の分類にはデータスパースネス問題が存在する [3]．この問題を解決するために，本研究では以下の特徴量および学習方法を利用することを提案する．

- アプリストアでは，アプリ名以外にもアプリの説明文やカテゴリ，インストール数などの様々な情報が存在するため，それらのマルチモーダルな情報を活用することが考えられる [4], [5]．特に，本研究ではローカルアプリを判定することを目的としているため，アプリ名や説明文における地名数を特徴量とすることを提案する．
- 近年，テキスト分類に深層学習の一種である Convolutional Neural Network（CNN）を適用することで分類性能の改善が行われている [6], [7], [8], [9]．そこでアプリ分類にテキストデータに対する CNN を適用することを提案する．
- 一般に，深層学習には大量の教師データが必要となる．画像認識の分野では ImageNet[10] など整備されたオープンデータがあるがアプリ分類に特化したデータはない．そこで，本研究ではアプリストアではアプリ提供者がアプリ公開時にカテゴリを手動で付与することに着目し，これを教師データとしてアプリ分類の事前学習（pre-training）を行い，ローカルアプリ分類には少数の教師データを用意し再学習（Fine-tuning）することで効率的にアプリ分類を行うことを提案する．

上記に加えて，深層学習はマルチモーダルな特徴量を組み合わせることに適している [11]．アプリストアのデータはテキスト，画像，カテゴリ，数値などの複数のモーダルの情報を持つため深層学習を適用するのに適していると考えられる．本研究では上記の 2 つの特徴量を深層学習で組み合わせて利用することを提案する．

本研究の貢献は以下の通りである．

- ローカルアプリ分類のため，アプリストアのデータからローカルアプリの特徴を考慮した特徴量（アプリ名

[1] 株式会社 NTT ドコモ, NTT DOCOMO, INC.
[a)] ochiaike@nttdoomo.com
[*1] https://play.google.com/store
[*2] https://www.apple.com/jp/ios/app-store/

© 2018 Information Processing Society of Japan

Vol.2018-DPS-175 No.9
Vol.2018-MBL-87 No.9
Vol.2018-ITS-73 No.9
2018/5/24

や説明文における地名数，カテゴリ，ダウンロード数など）を提案した．

- アプリ説明文に対して CNN を利用した特徴抽出を，アプリ提供者が付与したカテゴリで事前学習を行うことで効率的に学習することを提案した．
- アプリストアから生成した特徴量とアプリ説明文から CNN により特徴抽出した特徴量を統合しローカルアプリを判定する深層学習モデルを提案した．
- Google Play の実データを用いて評価実験を行い提案手法の有効性を確認した．

本稿では，次章にてアプリ分類に関する関連研究について述べ，本研究との差分について説明する．次に，3 章でアプリストアで提供されているデータおよびローカルアプリの定義を述べる．4 章にて提案手法の詳細を説明し，5 章では実データを用いた実験を行い，既存手法と提案手法の精度について検証する．最後に 6 章にて本研究のまとめおよび今後の課題について述べる．

2. 関連研究

アプリのカテゴリ分類に関する研究がいくつか行われている．Zhu らは，アプリのカテゴリ分類をアプリ名や，検索エンジンでアプリ名をクエリとして検索したときのスニペットを文書と見なし，テキスト分類することでカテゴリ分類を行った [1]．その際に，1 章で述べたように文書が短文となりデータスパースネス問題が生じる．そこで，単語の出現回数だけでなく，検索スニペットに対して LDA（Latent Dirichlet Allocation）[12] を適用し，文書のトピックを特徴量とすることでデータスパースネス問題の解決を図っている．スニペットごとに，単語の出現回数を特徴ベクトルに各カテゴリとのコサイン類似度を計算しもっとも類似度が高いカテゴリの頻度も特徴量として利用している．また，アプリの実世界での利用状況（時刻や曜日，位置情報など）を取得し，実世界での利用状況を特徴量として加えている．最終的に，アプリ名やスニペットから生成した特徴量（単語出現回数，カテゴリ頻度，LDA トピック分布）および実世界での利用状況から生成した特徴量を利用し最大エントロピーモデルによって予め定義したカテゴリにアプリを分類した．Li らは，Zhu らと同様のテキスト特徴量を利用し，ナイーブベイズ分類器を学習しアプリのカテゴリ分類を行った [2]．

Chen らはモバイルアプリに対して，アプリストアの情報をもとにタグ付けを行う手法を提案した [5]．この研究では，アプリ名，アプリ説明文，ユーザの評価，アプリのスクリーンショットなど 10 種類のモダリティのデータを特徴量としてアプリにタグ付けを行っている．教師データとしてタグが付与されているアプリとタグ付け対象のアプリで前述の特徴量からアプリ間類似度 [5] を計算しタグを付与する．



図 1 アプリストアのデータの例

アプリ分類の先行研究をまとめると，アプリ名や検索スニペットを文書としたテキスト分類問題としてカテゴリの分類を行う研究とアプリストアの複数の情報からアプリにタグ付けする研究が行われている．本研究のように，アプリストアの複数の情報をもとにカテゴリ分類を行う研究は行われていない．

3. アプリストアのデータとローカルアプリの定義

3.1 アプリストアのデータ

図 1 にアプリストアのデータの例を示す [*3]．アプリの名称や説明文，スクリーンショットなどなどが情報として提供されている．この中で，本研究ではアプリ名，カテゴリ，説明文，インストール数，更新日から特徴量を生成する．

3.2 用語定義

Point-of-Interest（POI） 人々が目的地や目標物とする店舗や施設，ランドマークなどの地理的な実体（geographic entity）を POI と定義し，名称，位置（緯度，経度）により表現される [13]．

エリア 都道府県，市町村などの行政区分や複数の緯度経度の点で指定される範囲で定められた地理的な実体と定義する．名称，地理的な範囲，代表位置（緯度，経度）で表現される [14]．

ローカルアプリ 本研究では POI，都道府県や市町村など名称と緯度経度で定義される地理的な実体に関する情報を提供するアプリをローカルアプリと定義する．ローカルアプリの例としては，POI に関する情報を提供するアプリ（例：横浜ワールドポーターズアプリ [*4]，

*3 https://play.google.com/store/apps/details?id=net.goyah.app
*4 https://play.google.com/store/apps/details?id=com.ywpapp

© 2018 Information Processing Society of Japan

仙台市博物館 [*5]），特定の都道府県や市区町村など自治やエリアに関する情報を提供するアプリ（例：横浜市ごみ分別アプリ [*6]，函館イベント情報「gocco」[*7]），POI のカテゴリに該当するアプリ（例：Golf Navi（ゴルフナビ） EAGLE VISION[*8]，山旅ロガー [*9]）などがある．

4. 提案手法

本章では，提案する特徴量の生成方法および分類器の学習方法について説明する．提案手法では，主に以下の 2 種類の特徴量を利用する．

(1) アプリストアの様々なモダリティから人手で設計した特徴量を生成する．
(2) アプリ説明文から CNN を用いて特徴量を生成する．

以降では，各特徴量の生成方法について詳しく説明する．なお，本研究では Google Play のデータを用いて評価を行ったため Google Play のデータに対する特徴量生成方法を説明する．

4.1 アプリストアの各種データからの特徴量生成（Multi-modal feature）

本研究で利用する特徴量の一覧を表 1 に示す．ローカルアプリではアプリ名や説明文に地名が多く出現するという仮説のもと，アプリ名や説明文に含まれる都道府県名，市町村名，POI 名の出現数を特徴量とする．アプリのカテゴリについては，ローカルアプリそのものを示すカテゴリは存在しないものの，旅行などのカテゴリはローカルアプリに近いと考えられるためカテゴリを特徴量とする．インストール数は，全国的に利用されるアプリでは利用者数が多く，特定の地域で利用されるアプリはそれより利用者が少ないという仮説で採用した．Google Play では「1000〜5000」「10000〜50000」のように区間ごとのどれに該当するかがデータとして与えられている．そのため，この区間をカテゴリ変数として特徴量にした．更新年は，ローカルアプリは比較的新しい年にリリースされたという仮説のもと採用した．

4.2 CNN による特徴抽出（CNN feature）

アプリの説明文からカテゴリに分類に有効な特徴量を抽出するため，本研究では Convolutional Neural Network（CNN）を利用する．一般に，深層学習には大量の教師データが必要となるため，ドメインごとにパブリックに利用できるデータセットで事前学習（Pre-training）を行い，各タスクごとに再学習（Fine-tuning）することで高性能な分類器を学習する．例えば，画像認識の分野では ImageNet の大量の画像で事前学習し，物体認識などの個別のタスク用に用意した教師データで再学習する研究がある [15]．しかしながら，アプリ分類に利用できるような言語データのオープンデータは存在しない．そこで， 本研究ではアプリストアではアプリ提供者がアプリ公開時にカテゴリを手動で付与することに着目した．例えば，Google Play では「ライフスタイル」「地図&ナビ」「ソーシャル」などのカテゴリをアプリ提供者が Google Play にアプリを登録するときに選択する [*10]．これをカテゴリの教師データと見なすことで，アプリストアから大量の教師ありデータを取得することができる．本研究では，Google Play のカテゴリを説明文から推定するタスクを事前学習として行い，ローカルアプリの分類を説明文から推定するタスクを再学習としてニューラルネットのパラメータを学習する．

表 1 提案手法で利用する特徴量一覧

元データ	特徴量
アプリ名	含まれる都道府県名数
アプリ名	含まれる地名数（市町村名、POI 名）
説明文	含まれる都道府県名数
説明文名	含まれる地名数（市町村名、POI 名）
カテゴリ	カテゴリ
インストール数	インストール数（カテゴリ変数）
更新年	更新年

図 2(a) に本研究で利用する CNN の構造を示す．最初に Embedding Layer で説明文（description）を分散表現に変換し，その後，3 層の Convolution Layer と 3 層の Max Pooling Layer を利用する．そして，Flatten Layer で平坦化し，Full Connect Layer を経由してカテゴリ分類を行う．Full Connect Layer では，Dropout[16] を用いて汎化性能を高める．Embedding Layer の初期値には，Word2vec[17] や fasttext[18] などで，公開データで学習された単語の分散表現を利用することができる．Convolution Layer では活性化関数として ReLU（Rectified Linear Unit）[19]，Full Connect Layer の活性化関数には Softmax 関数を利用する．Softmax 関数を用いることで，各ユニットの出力を各ラベルの所属確率とすることができる．ここで，出力層におけるユニット数を N，入力を x，ユニット i の出力を x_i とした時，ユニット i の出力 p_i は式（1）で定義される．

$$p_i = \frac{e^{x_i}}{\sum_{j=1}^{N} e^{x_j}} \tag{1}$$

事前学習では，Google Play における各アプリのカテゴリを予測するため，N はカテゴリ数となり，カテゴリ数分のクラス分類問題を行う．

分類器の損失関数 L_{CLF} には cross-entropy を用いる．サンプル数を n，分類クラス数を m，サンプル i における

[*5] https://play.google.com/store/apps/details?id=jp.co.isb.itc.museum
[*6] https://play.google.com/store/apps/details?id=jp.co.haleng.yokohamagomi
[*7] https://play.google.com/store/apps/details?id=com.ivg.hakodateevent
[*8] https://play.google.com/store/apps/details?id=jp.co.asahigolf.eva.pro
[*9] https://play.google.com/store/apps/details?id=com.kamoland.ytlog
[*10] https://support.google.com/googleplay/android-developer/answer/113475?hl=ja

© 2018 Information Processing Society of Japan

Vol.2018-DPS-175 No.9
Vol.2018-MBL-87 No.9
Vol.2018-ITS-73 No.9
2018/5/24

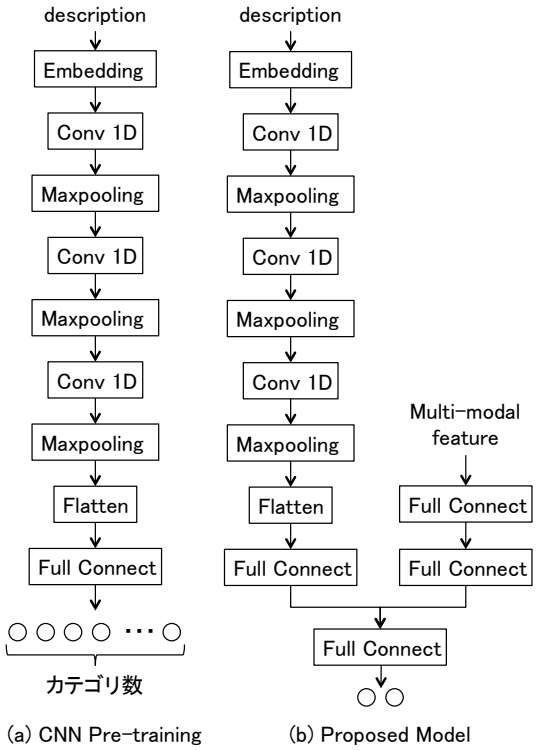


図 2 提案する深層学習のアーキテクチャ

クラス j の分類器出力を p_{ij} としたとき，損失関数 L_{CLF} は式（2）で定義される．

$$L_{CLF}(\boldsymbol{w}) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{m} y_{ij} log(p_{ij}) \qquad (2)$$

ニューラルネットワークの重み $\boldsymbol{w}$ は式（2）の誤差を確率的勾配降下法（Stochastic Gradient Descent，SGD）[20] を利用し最小化することで学習する．

4.3 分類器の構築（再学習：Fine-tuning）

本節では，アプリストアの各種データから生成した特徴量と CNN で生成した特徴量からローカルアプリを分類するための分類器の学習について説明する．提案する分類器のニューラルネットワークの構造を図 2(b) に示す．左側の CNN の部分では，前節で述べた事前学習により学習した重みを初期値として利用する．右側の部分は 4.1 節で生成した特徴量を入力し，Full Connect Layer を 2 層経由する．そして，CNN で生成した特徴量とアプリストアの各種データから生成した特徴量を連結し，さらに Full Connect Layer を経由して，ローカルアプリか否かの 2 値分類を行う．Convolution Layer では活性化関数として ReLU，Full Connect Layer の活性化関数には最終層では Softmax 関数を，それ以外の層では ReLU を利用する．分類器のニューラルネットワークの重み $\boldsymbol{w}$ は，事前学習と同様に損失関数に cross-entropy を設定し，確率的勾配降下法により学習する．

表 2 事前学習に利用したアプリカテゴリとアプリ数

カテゴリ	アプリ数	カテゴリ	アプリ数
エンタメ	15016	ソーシャルネットワーク	2275
カスタマイズ	14979	ビジネス	2095
ツール	12638	スポーツ	1588
ライフスタイル	12181	ファイナンス	1503
教育	10405	通信	1435
写真	4968	ショッピング	1410
音楽＆オーディオ	4281	医療	1408
旅行＆地域	3478	書籍＆文献	1385
仕事効率化	3324	ニュース＆雑誌	1343
健康＆フィットネス	2985	メディア＆動画	1303

5. 評価実験

本章では提案手法の有効性を確認するために行った評価実験について説明する．はじめに，実験で利用したデータや機械学習モデルのパラメータ設定，実行環境などについて述べる．次にベースラインとした比較手法と評価指標について説明する．その後，定量評価の結果を説明する．

5.1 実験条件およびデータ

評価のため，2016 年 6 月 28 日から 2017 年 3 月 11 日までに Google Play で公開されたアプリの情報を約 40 万件取得した（以下，GP データと呼ぶ）．GP データ中で，学習データとして 1,518 件（正例，負例ともに 759 件）とホールドアウト評価用データとして 500 件（正例：84 件，負例：416 件）に対して人手でラベル付けを行った．評価用データで負例を多くしているのは，実環境におけるデータの比率は正確にはわからないが，実環境ではローカルアプリよりその他のアプリの方が多いと考えられるためである．4.2 節で説明した CNN の事前学習には，GP データ中で登録数の多いカテゴリの上位 20 位までのアプリを約 10 万件利用した．この約 10 万件のデータで 20 クラス分類を行うことで事前学習した．表 2 に事前学習に利用したアプリカテゴリとアプリ数を示す．

CNN では入力の次元数を固定する必要があるため，各アプリの説明文から先頭 1,000 単語までを入力に利用した．単語の次元数は 300 とした．CNN の特徴マップの数は 128，フィルタサイズは 5 とした．

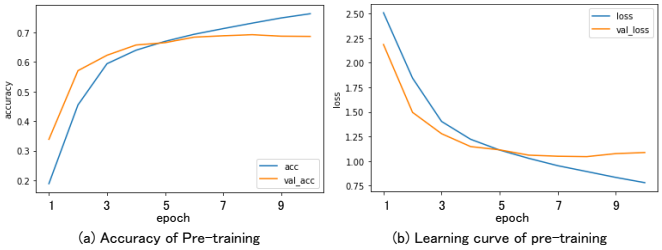
実装環境は Ubuntu 16.04，Python 3.6 を利用し，深層学習の実装には Tensorflow[21] および Keras[*11] を利用した．後述するベースライン手法および提案手法において，機械学習モデルのパラメータを複数設定し検証を行い最も性能が良いパラメータで比較を行った．

5.2 比較手法

本研究では，類似研究における最新の研究として Li らの研究 [2] と同様の手法を実装し比較を行った．Li らの研

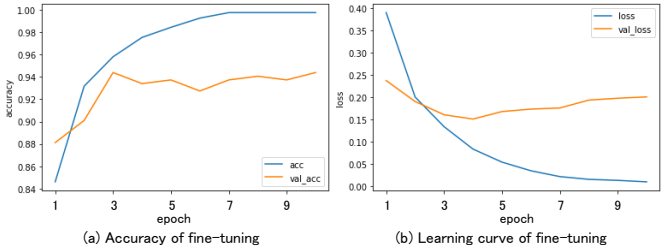
[*11] https://keras.io/

© 2018 Information Processing Society of Japan



(a) Accuracy of Pre-training　(b) Learning curve of pre-training

図 3　事前学習における Accuracy と loss



(a) Accuracy of fine-tuning　(b) Learning curve of fine-tuning

図 4　再学習における Accuracy と loss

究では以下の 2 つの特徴量を利用している．
(1) 単語の出現頻度に基づく特徴量
(2) 文書のトピックに基づく特徴量

前者は文書分類でよく用いられる特徴量である．後者は単語の出現のみを特徴量とすると意味が似ている単語でも別の特徴となるため，まったく同一の単語でなくとも意味が類似している単語の場合は特徴量に加えることを目的としている．具体的には，文書のトピック（カテゴリ）の所属確率を特徴量として利用する．実装では，単語の出現頻度に基づく特徴量として TF-IDF[22] を利用し，文書のトピックに基づく特徴量として各文書の LDA のトピック分布を利用する．

5.3　評価指標

実験の定量的な評価指標には再現率（recall），適合率（precision）および F 値（F-measure）を利用する．再現率は人手でローカルアプリとラベルを付けたアプリのうち，いくつのアプリを抽出できたかという割合，適合率は各手法でローカルアプリと判定したアプリのうち，人手で正解ラベルを付与したアプリの割合である．F 値は再現率と適合率の調和平均によって求められる．計算方法は以下の通り．

$$\text{F 値} = \frac{2 \times \text{再現率} \times \text{適合率}}{\text{再現率} + \text{適合率}}$$

5.4　評価結果と考察

図 3（b）に事前学習の学習曲線を示す．事前学習におけるテストデータに対する損失（loss）およびバリデーションデータに対する loss（val_loss）が学習回数（epoch 数）ごとに少なくなっていることがわかる．図 3（a）に事前学習における Accuracy を示す．学習が進むに従い，精度が向上していることがわかる．また，図 4（a）に再学習の Accuracy，図 4（b）に再学習の学習曲線を示す．この図から再学習において学習が進むに従い，精度が向上し，loss が低下していることがわかる．損失については，学習回数 5 程度から数値が上昇しているため，5epoch で打ち切った方がよいと考えられる．

定量評価の結果を表 3 に示す．太字は最も数値が高いことを示す．Multi-modal feature + RF は 4.1 節で説明した Multi-modal feature を特徴量とし Random Forest で分類したことを示す．同様に Multi-modal feature + MLP は Multi-modal feature を特徴量とし多層パーセプトロン（Multi Layer Perceptron）で分類したことを示す．CNN feature（事前学習なし）は，図 2(a) の最終層で 2 値分類を行う CNN モデルで Google Play のカテゴリ分類による事前学習は行わず学習データのみで重みを計算した場合を示す．CNN feature（事前学習あり）は，図 2(a) の最終層で 2 値分類を行う CNN モデルで Google Play のカテゴリ分類による事前学習を行った後に，ローカルアプリを分類する 2 値分類を最終層として再学習した場合を示す．Proposed Model は図 2(b) に示すように Multi-modal feature と CNN feature の両方を利用し，CNN feature については事前学習を行った．

表 3　各手法での評価結果

手法	適合率	再現率	F 値
Li et al. (2016) [2]	0.884	0.868	0.874
Multi-modal feature + RF	0.907	0.878	0.887
Multi-modal feature + MLP	0.916	0.902	0.907
CNN feature（事前学習なし）	0.703	0.582	0.629
CNN feature（事前学習あり）	**0.930**	0.892	0.901
Proposed Model	**0.930**	**0.928**	**0.929**

表 3 より，提案手法のモデルが全ての指標において最も高性能に分類することができた．人手で設計した Multi-modal feature を利用した場合において，ベースライン手法より各指標で約 1 %〜3 %程度の性能を改善した．CNN feature では事前学習した場合，ベースライン手法より高性能に分類を行うことができた．CNN feature の事前学習ありとなしの比較から，Google Play のカテゴリ分類によって事前学習する効果があることがわかる．

6.　おわりに

本研究では，アプリストアで公開されているアプリ説明文から抽出した地名数やカテゴリ，インストール数をなどの Multi-modal 特徴量，アプリ説明文から CNN を用いて抽出した CNN 特徴量を用いてローカルアプリを分類する手法を提案した．CNN 特徴量を生成する際，アプリストアの各アプリはアプリ提供者によりカテゴリを付与されていることに着目し，アプリストアの大量のアプリを用いて事前学習することを提案した．Google Play のアプリデータを利用し評価を行い，提案手法がベースライン手法より

© 2018 Information Processing Society of Japan

高精度にローカルアプリを分類できることを確認した．

今後の課題として，アプリの利用ログと組み合わせて分類することや，ローカルアプリ以外へのアプリ分類への応用が考えられる．

参考文献

[1] Zhu, H., Chen, E., Xiong, H., Cao, H. and Tian, J.: Mobile app classification with enriched contextual information, *IEEE Transactions on mobile computing*, Vol. 13, No. 7, pp. 1550–1563 (2014).

[2] Li, X., Lian, Y.-h. and Yu, H.: Classification of mobile APPs with combined information, *Cloud Computing and Big Data Analysis (ICCCBDA), 2016 IEEE International Conference on*, IEEE, pp. 193–198 (2016).

[3] Phan, X.-H., Nguyen, L.-M. and Horiguchi, S.: Learning to Classify Short and Sparse Text & Web with Hidden Topics from Large-scale Data Collections, *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, New York, NY, USA, ACM, pp. 91–100 (online), DOI: 10.1145/1367497.1367510 (2008).

[4] Chen, N., Hoi, S. C., Li, S. and Xiao, X.: SimApp: A Framework for Detecting Similar Mobile Applications by Online Kernel Learning, *Proceedings of the Eighth ACM International Conference on Web Search and Data Mining*, WSDM '15, New York, NY, USA, ACM, pp. 305–314 (online), DOI: 10.1145/2684822.2685305 (2015).

[5] Chen, N., Hoi, S. C., Li, S. and Xiao, X.: Mobile App Tagging, *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*, WSDM '16, New York, NY, USA, ACM, pp. 63–72 (online), DOI: 10.1145/2835776.2835812 (2016).

[6] Kim, Y.: Convolutional Neural Networks for Sentence Classification, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Association for Computational Linguistics, pp. 1746–1751 (online), DOI: 10.3115/v1/D14-1181 (2014).

[7] Severyn, A. and Moschitti, A.: Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks, *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, New York, NY, USA, ACM, pp. 373–382 (online), DOI: 10.1145/2766462.2767738 (2015).

[8] Severyn, A. and Moschitti, A.: Twitter Sentiment Analysis with Deep Convolutional Neural Networks, *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, New York, NY, USA, ACM, pp. 959–962 (online), DOI: 10.1145/2766462.2767830 (2015).

[9] Lee, K., Qadir, A., Hasan, S. A., Datla, V., Prakash, A., Liu, J. and Farri, O.: Adverse Drug Event Detection in Tweets with Semi-Supervised Convolutional Neural Networks, *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, Republic and Canton of Geneva, Switzerland, International World Wide Web Conferences Steering Committee, pp. 705–714 (online), DOI: 10.1145/3038912.3052671 (2017).

[10] Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K. and Fei-Fei, L.: Imagenet: A large-scale hierarchical image database, *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, IEEE, pp. 248–255 (2009).

[11] Zhang, S., Yao, L. and Sun, A.: Deep learning based recommender system: A survey and new perspectives, *arXiv preprint arXiv:1707.07435* (2017).

[12] Blei, D. M., Ng, A. Y. and Jordan, M. I.: Latent dirichlet allocation, *Journal of machine Learning research*, Vol. 3, No. Jan, pp. 993–1022 (2003).

[13] Rae, A., Murdock, V., Popescu, A. and Bouchard, H.: Mining the Web for Points of Interest, *Proceedings of the 35th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '12, pp. 711–720 (2012).

[14] 国土地理院：地理情報標準第 2 版（JSGI 2.0），国土地理院（オンライン），入手先 ⟨http://www.gsi.go.jp/GIS/stdind/jsgi2.html⟩（参照 2018-03-06）.

[15] Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S. and Darrell, T.: Caffe: Convolutional architecture for fast feature embedding, *Proceedings of the 22nd ACM international conference on Multimedia*, ACM, pp. 675–678 (2014).

[16] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research*, Vol. 15, No. 1, pp. 1929–1958 (2014).

[17] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J.: Distributed representations of words and phrases and their compositionality, *Advances in neural information processing systems*, pp. 3111–3119 (2013).

[18] Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T.: Enriching word vectors with subword information, *arXiv preprint arXiv:1607.04606* (2016).

[19] Glorot, X., Bordes, A. and Bengio, Y.: Deep Sparse Rectifier Neural Networks., *Aistats*, Vol. 15, No. 106, p. 275 (2011).

[20] Bottou, L.: Large-scale machine learning with stochastic gradient descent, *Proceedings of COMPSTAT'2010*, Springer, pp. 177–186 (2010).

[21] Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M. et al.: TensorFlow: A System for Large-Scale Machine Learning., *Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283 (2016).

[22] Manning, C. D., Raghavan, P. and Schütze, H.: *Introduction to Information Retrieval*, Cambridge University Press, New York, NY, USA (2008).

© 2018 Information Processing Society of Japan