文章類似度に基づくユーザーリコメンドシステムの構築

○竹井　優太，佐藤　浩，白川　智弘（防衛大学校）

概要　既存のユーザーリコメンドシステムには，活動履歴の少ないユーザーに対するリコメンド，ユーザー評価の少ない新規のコンテンツのリコメンドをしにくいという問題や，小規模なサービスでは適切なリコメンドをしにくい，などの問題がある．このような問題を解決するため，過去の研究において，ユーザー同士の文章類似度に基づきリコメンドを行う Document Vector Matching (DVM) と名付けられた手法が提案された．DVM では，ユーザーの投稿した文章をベクトル化し，ベクトル間の距離が近く，文章類似度が高いユーザーペアをマッチングする．本研究では，過去の研究を踏襲し，ユーザーの活動履歴やサービスの規模に依存しない，文章類似度に基づくユーザーリコメンドシステムの構築を行う．

キーワード：リコメンドシステム，文章類似度，自然言語処理，DVM

1　はじめに

近年，自然言語処理に基づくリコメンドは，商業やエンターテイメント等の分野において盛んに行われており[1]，身近な例ではグルメ，トラベルそしてショッピングなどのオンラインサービスが挙げられる．既存のリコメンドシステムにおいては内容ベースフィルタリングや，協調フィルタリングといった手法が主流である[2]．内容ベースフィルタリングとは個人属性を基にリコメンドをすることで，例えば対象者（リコメンドを受けるユーザー）が過去に購入した商品と似たような商品をリコメンドすることが挙げられる．協調フィルタリングとは標本利用者の嗜好データを基にリコメンドをすることで，例えば「この商品を買った人はこんな商品も買っています」といったリコメンドが挙げられる．

これらのリコメンドシステムには活動履歴の少ないユーザーが対象者になりにくい（コールドスタート問題），小規模なサービスだと適切なリコメンドをしにくいといった弱点がある[2]．これらの弱点を克服するべく渡邉らは，ユーザーが投稿した文章に基づくリコメンドシステムを提案した．即ち，ユーザー間の文書類似度に基づきリコメンドすることにより，サービスの規模やユーザーの購買・閲覧履歴の蓄積に依存しないシステムの構築を目指した[3]．

渡邉らは BoW[4] や Doc2Vec[5]，BERT[6] といった自然言語処理モデルを用いてユーザーの投稿文章をベクトル化し，ベクトル同士の Cos 類似度やユークリッド距離を測ることで文章類似度を求めた．そして，Twitter において文章類似度とフォロー関係の相関を分析した結果，ユーザーは自分の文章と似ている文章を投稿するユーザーを好む傾向があることを明らかにした．これは文章類似度のみにより，ある一定の精度でリコメンドを行える可能性があることを意味する．

この実験で，文章類似度とフォロー関係の相関が最も強く結果に出たのは，ユーザーの投稿した文章を Doc2Vec によりベクトル化し，その Cos 類似度を文章類似度としたときであった．このことから渡邉らは，この組み合わせで計算した文書類似度を基にユーザー同士をマッチングさせる Document Vector Matching (DVM) を提案した (Fig. 1).

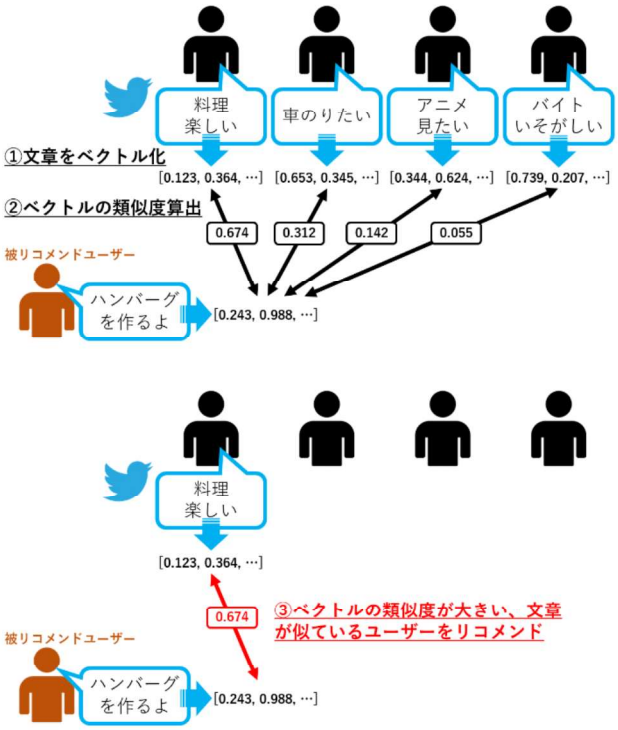

Fig. 1: Mechanism of DVM.

しかし実験では，ベクトル化や距離計算の手法として限定的なものしか用いられておらず，また，DVM がリコメンドするうえで役に立つ可能性を示唆したのみで，実際のリコメンドにおける有効性は検証されていない．そのため DVM に用いる文章のベクトル化手法とその距離算出方法の組み合わせにはまだ検討の余地があり，さらに，有効性を実際に実験することにより示す必要がある．

本研究では，渡邉らの研究を踏襲し，ユーザーの活動履歴やサービスの規模に依存しない，文章類似度に基づくユーザーリコメンドシステムの構築を行う．実験では，文章をベクトル化するための自然言語処理モデルに Word2Vec[7], fastText[8], Doc2Vec, BERT, RoBERTa[9], ALBERT[10], DistilBERT[11], ELECTRA[12] を，ベクトル間の距離には Cos 類似度，ユークリッド距離，マンハッタン距離，チェビシェフ距離，マハラノビス距離を用いる．そして，これらの組み合わせで計

21PG0002/21/0000-0040

算したユーザー同士の文章類似度とそのフォロー関係の相関を分析し，どの組み合わせを用いて文章類似度を計算すれば最も効率よく類似したユーザーを抽出できるかを探る．その後，最も良い結果を示した組み合わせを DVM に組み込んだユーザーリコメンドシステムを構築し，Twitter 上で実際にどれだけエンゲージメントが発生するのか実験を行い，DVM の有効性を示す．

2 予備知識

2.1 リコメンドアルゴリズム

リコメンドアルゴリズムには大きく 4 種類存在し，それぞれに長所と短所が存在する．

2.1.1 コンテンツベースフィルタリング

コンテンツベースフィルタリングとは，コンテンツの属性を基に，類似するコンテンツをリコメンドするアルゴリズムである．ここで言う属性とは，各コンテンツに対し付与されたタグやコンテンツのユーザー評価，コンテンツの説明文などのことである．レストラン A を訪れたユーザーに，同じようなタグが付いているレストラン B をリコメンドする，映画 A を観たユーザーに，映画 A の説明文と類似する説明文をもつ映画 B をお勧めするなどのリコメンドが，コンテンツベースフィルタリングの例として挙げられる．

2.1.2 協調フィルタリング

協調フィルタリングには，アイテムベースとユーザーベースの 2 種類がある．アイテムベースとは，ユーザーの活動履歴をもとにコンテンツ同士の類似度を算出し，その蓄積データをもとにリコメンドするアルゴリズムである．多くのユーザーの動向から商品 A と商品 B の類似度が高いことがわかったため，商品 A を購入したユーザーには商品 B をお勧めするなどのリコメンドが，アイテムベースの例として挙げられる．ユーザーベースとは，ユーザーの活動履歴をもとにユーザー同士の類似度を算出し，その蓄積データをもとにリコメンドするアルゴリズムである．ユーザー A と B の動向から両者の類似度が高いことが分かったため，ユーザー A が購入した商品をユーザー B にもお勧めするなどのリコメンドが，アイテムベースの例として挙げられる．

2.1.3 コンテンツベースフィルタリングと協調フィルタリングの問題点

コンテンツベースフィルタリングと協調フィルタリングが抱える問題として，コールドスタート問題がある．コールドスタート問題とは，活動履歴の少ないサービスの新規ユーザーに対して適切なリコメンドをしにくい，また，ユーザー評価の少ない新規のコンテンツをリコメンドしにくいという問題である．このほかにも，小規模なサービスの場合，蓄積データが限られたものとなり適切なリコメンドをしにくい，マイナーなコンテンツは蓄積データのみからではリコメンドしにくいといった問題も存在する．

2.2 SNS

本研究で実験に使用する SNS は，ユーザーの投稿した文章データを収集し，その後，DVM の有効性を実際に検証するという目的から，投稿型の SNS でなければならない．そこで，投稿型の SNS の中でも，利用者が特に多く，ユーザー同士を比較しやすい，また，ユーザーデータを収集できるインターフェースを兼ね備えている Twitter, Facebook, Instagram を実験に使用する SNS の候補とした．

そして，この 3 つのサービスを，文章データの収集のしやすさやインターフェースの利用のしやすさから比較した結果，Twitter を本研究に用いることとした．

2.2.1 Twitter

Twitter の国内ユーザー数は約 4, 500 万人である．このサービスは短い文章の投稿を通してユーザーの嗜好にあうコンテンツを追いかけたり，発信したりすることをコンセプトとする．Twitter API というインターフェースを用いれば，ツイート，いいね，リツイートやフォローなどの Twitter のサービスに，公式 Web サイトを通さずにアクセスできる．

渡邉らは，Twitter のユーザー数，投稿されるテキストの量の豊富さやTwitter API の簡便性などを考慮し，研究にツイートのデータを用いた[3]．本研究でも引き続き，検証実験は Twitter を用いて行う．

2.2.2 Facebook

Facebook の国内ユーザー数は約 2, 600 万人である．このサービスは投稿，写真や動画の投稿を通して，友人や家族とコンテンツを共有することをコンセプトとする．

かつては，Facebook API を用いて，Facebook のサービスに公式 Web サイトを通さずにアクセスできた．しかしながら，プライバシー保護の観点から 2019 年に API の権限が縮小されたため，現在では大規模なデータ解析を行うのが難しくなっている．そのため本研究では Facebook を実験に使用する SNS の候補から除外した．

2.2.3 Instagram

Instagram の国内ユーザー数は約 3, 300 万人である．このサービスは写真や動画の投稿を通してユーザーの嗜好にあうコンテンツを追いかけたり，発信したりすることをコンセプトとする．国内では「インスタ映え」が 2017 年に流行語大賞に選ばれるなど，写真の投稿が近年益々人気を博している SNS である[13]．

Instagram API は提供元である Facebook のプライバシー保護の方針に従い，その権限を縮小されたため，大規模なデータ解析を行うのが難しい．そのため，本研究では Instagram を実験に使用する SNS の候補から除外した．

2.3 自然言語処理モデル

近年，自然言語処理モデルは研究が盛んであり，State-of-the-Art model（最も性能が良いモデルのこと，以下 SOTA）が頻繁に更新されているほか，様々な言語で様々なモデルがファインチューニングされ公開

されている．しかし，渡邉らの研究で扱った自然言語処理モデルは BoW, Doc2Vec, BERT の 3 種類のみで，種類が少ないうえに，モデルも古い．

そこで，本研究では Word2Vec 以降の機械学習による自然言語処理モデルで，公開されている日本語の学習済みモデルをできるだけ数多く探し，Word2Vec, fastText, Doc2Vec, BERT, DistilRoBERTa, ALBERT, DistilBERT, ELECTRA を実験に使用することとなった．

2.3.1 Word2Vec

Word2Vecとは単語の埋め込み表現，すなわち，単語ベクトルを生成するための自然言語処理モデル群の総称であり，2013 年 1 月 16 日に Google から発表された[7]. Word2Vec には CBoW (Continuous Bag-of-Words) モデルと Skip-gram (Continuous Skip-gram) モデルの 2 種類あり，それぞれ学習方法が異なる．

CBoW モデルは教師あり学習で，周辺の単語の One-hot ベクトルを入力として与えて，予測される中心の単語を出力し，ネットワークにある単語の中心にどのような単語が現れるのかを学習する (Fig. 2).

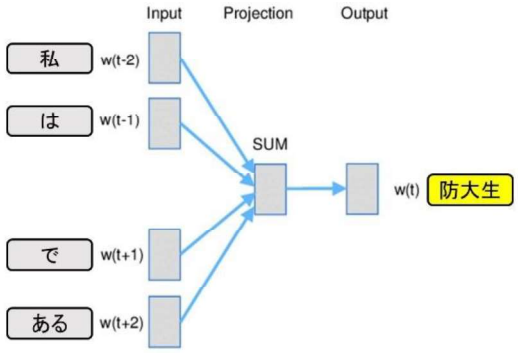

Fig. 2: Mechanism of CBow[14].

Skip-gram モデルも教師あり学習で，CBoW とは逆に，中心の単語の One-hot ベクトルを入力として与えて，予測される周辺の単語を出力し，ネットワークにある単語の周辺にどのような単語が現れるのかを学習する (Fig. 3).

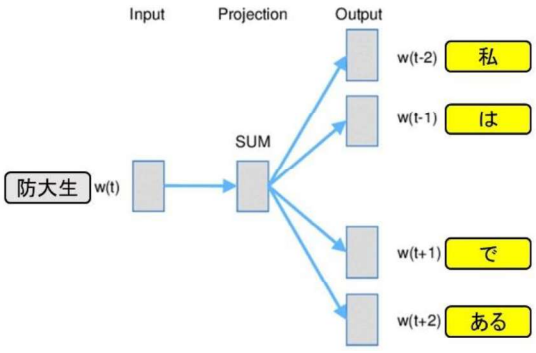

Fig. 3: Mechanism of Skip-gram[14].

One-hot ベクトルとはあるクラスのベクトルを対応要素だけ 1 とするベクトルで表現することである．例えば Fig. 4 のように「私」「僕」「は」「が」「防大生」「東大生」の 6 つのクラスがある場合，単語である「防大生」の One-hot ベクトルは (0, 0, 0, 0, 1, 0) となり，文章である「僕は東大生」の文章 ID は，単語である「僕」,「は」と「東大生」のクラスのインデックスを並べた (“236”) となる．

Fig. 4: One-hot vector and sentence ID.

Word2Vec では CBow と Skip-gram のいずれかの学習方法を用いて，文中の単語の出現確率を測定することで単語ベクトルを生成している．

本研究では https://github.com/singletongue/WikiEntVec に公開されている，日本語版 Wikipedia のコーパス（大量の文章の集合）を Skip-gram により学習し，300 次元の単語ベクトルを生成するモデルを用いた．

2.3.2 fastText

fastText とは Word2Vec の派生モデルであり，単語ベクトルを生成するための自然言語処理モデルである．2016 年 7 月 6 日に Facebook 人工知能研究所から発表された[8]. fastText は単語の n-gram （文章を連続する n 個の文字に分割すること）の埋め込み表現を考慮した Subword と呼ばれる仕組みを導入することにより，Word2Vec に比べ，より高速かつ高精度な単語ベクトルの生成を可能とした．

Subword は通常の単語により得られる埋め込み表現と，n-gram としての文字列の埋め込み表現を別物として扱う．例えば，「防大生」を 2-gram で分割すると「防大」と「大生」に分かれるが，Subword の処理下では，名詞単語としての「防大」と，「防大生」を 2-gram で分割した時に得られる 1 つ目の要素である「防大」は別の単語ベクトルを持つ (Fig. 5).

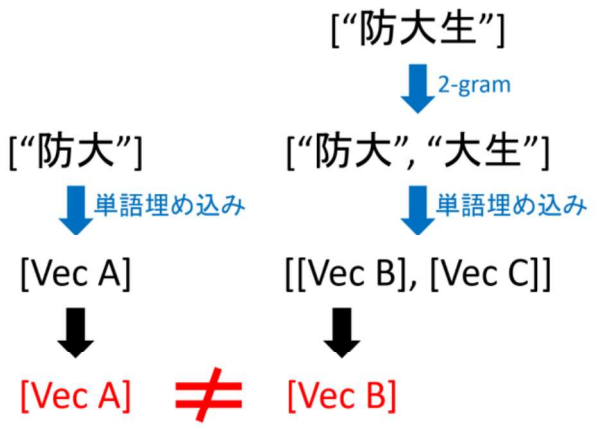

Fig. 5: Scheme of Subword.

fastText では n-gram を用いて未知の単語にも有効な埋め込みを学習させることができるという長所を持つ．しかしその特徴を持つがゆえに，ある単語が別の単語に含まれると両者のベクトルは類似する傾向があるため，「ティッシュ」の類似単語として「ブリティッシュ」が挙げられるなど，類似単語を見つけるなどの特定のタスクにおいては精度が落ちることが指摘されている．

本研究では https://github.com/Hironsan/awesome-embedding-models に公開されている，日本語版 Wikipedia のコーパスを Skip-gram により学習し，300 次元の単語ベクトルを生成するモデルを用いた．

2.3.3 Doc2Vec

Doc2Vec とは Word2Vec を発展させた，単語ベクトルに加え文章ベクトルの生成を可能にしたモデルであり，CBoW モデルの後継である PV-DM (Distributed Memory model of Paragraph Vector) モデルと，Skip-gram モデルの後継である PV-DBoW (Distributed Bag-of-Words version of Paragraph Vector) モデルの 2 種類がある．2014 年 5 月 16 日に Google から発表された[5]．

PV-DM モデルは，文章 ID と文章に含まれる単語の One-hot ベクトルを入力として与えて，予測される次の単語を出力し，文章の埋め込み表現を学習する (Fig. 6).

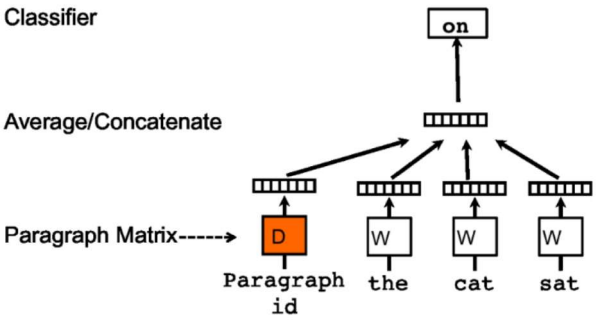

Fig. 6: Mechanism of PV-DM[5].

PV-DBoW モデルは，文章 ID を入力として与えて，同一文章に含まれる単語をサンプルとして用意した上で，サンプルに含まれる単語を予測して出力し，文章の埋め込み表現を学習する (Fig. 7).

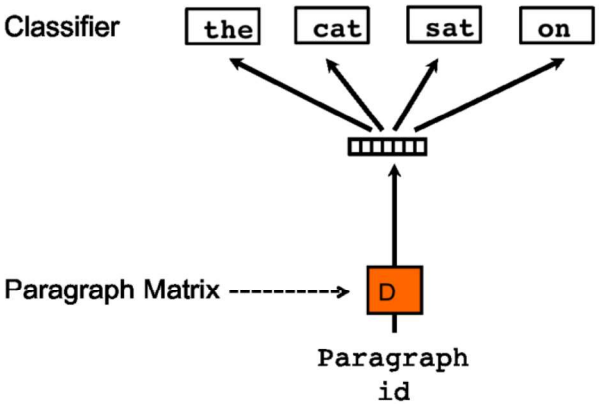

Fig. 7: Mechanism of PV-DBoW[5].

PV-DBoW は単語ベクトルを学習の際に用いる必要がないため PV-DM よりも学習コストが少ない．しかし PV-DBoW は学習で語順を無視するため PV-DM のほうがタスクの精度が高いといわれている．

本研究では https://github.com/yagays/pretrained_doc2vec_ja に公開されている，日本語版 Wikipedia のコーパスを PV-DM により学習し，300 次元の文章ベクトルを生成するモデルを用いた．

2.3.4 BERT

BERT とは Bidirectional Encoder Representations from Transformers の略で，翻訳，文章分類や質疑応答などの様々なタスクを行うための自然言語処理モデルである．2018 年 10 月 11日に Google から発表された[6]．

Transformer とは Attention と呼ばれる注意機構のみにより構成された深層学習モデルで 2017 年 6 月 12 日に Google から発表された[15]．Attention とは query によって memory (key や Value) から情報を引き出す仕組みで，辞書オブジェクトのようなものである．Fig. 8 では例として Query = “Key2” によりメモリーから Value2 を引き出している．

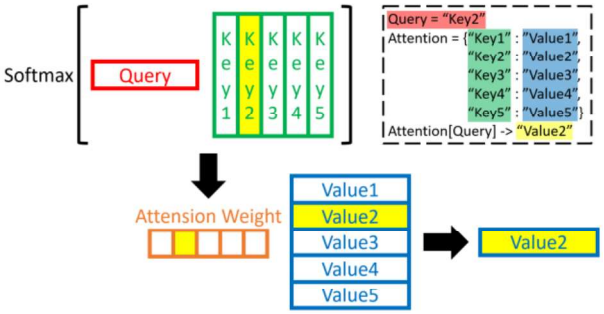

Fig. 8: Scheme of Attention.

Transformer は既存の RNN (Recurrent Neural Network) や CNN (Convolutional Neural Network) を使用したモデルとは違い，注意機構のみを使用し，再帰や畳み込みといった機構を一切使用していないため，並列して多くの計算をすることができるように

なった．そのため短い学習時間でより高い性能を実現している (Fig. 9).

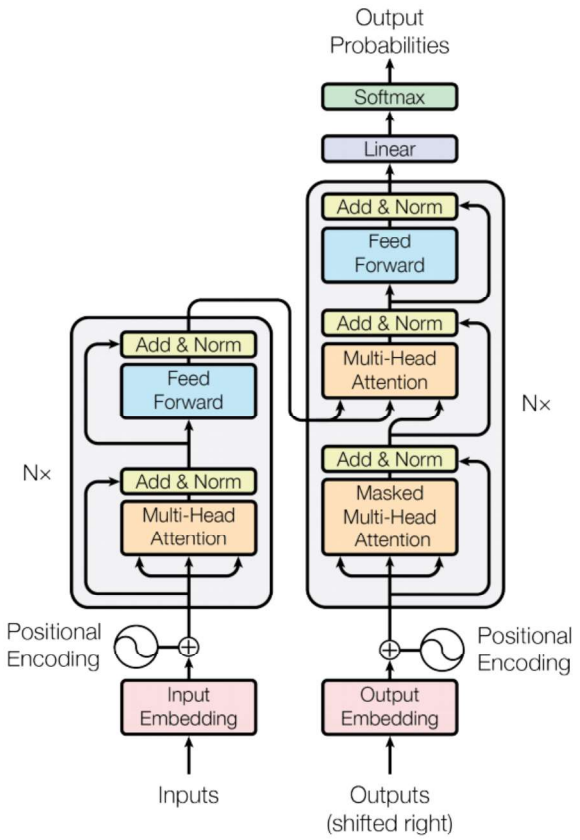

Fig. 9: Mechanism of Transformer[6].

BERT は従来単一方向からしか学習できなかった文章を，Transformer を用いて双方向から学習しているため，既存モデルに比べ自然言語処理タスクの精度が格段に良く，当時の各タスクの SOTA をことごとく塗り替えた．自然言語処理モデルの革新的モデルとなった BERT を基に，その発表後も ALBERT やRoBERTa といった派生モデルが次々と生まれた．

BERT は文章を双方向から学習しているため，文脈の理解に優れており，また，多様な自然言語処理タスクに応用できるため汎用性もある．さらに，既存モデルと違いラベルの付与されていないデータセットを処理することができるという強みがある．しかしながら，精度が高いのと引き換えに BERT の計算には多くのパラメータが必要で，学習に時間がかかる，パラメータを増加しても精度がある一定のレベルまでしかあがらないなどの弱点も存在する．

本研究では https://github.com/cl-tohoku/bert-japanese に公開されている日本語版 Wikipedia のコーパスを学習し，768 次元の単語ベクトルを生成するモデルと，https://github.com/UKPLab/sentence-transformers/blob/master/docs/pretrained_models.md に公開されている，50 ヵ国語以上の言語に対応した，768 次元の単語ベクトルを生成するマルチリンガルモデルを用いた．

2.3.5 RoBERTa

RoBERTa とは，BERT の派生モデルで，2019 年 7 月 26 日に Facebook 人工知能研究所から発表された[9]．RoBERTa は BERT の事前学習用データセットと学習の回数を増やしており，学習の方法においては，バッチサイズを大きくし，長い文章を投入するなどの改良を行っている．

本研究では https://github.com/UKPLab/sentence-transformers/blob/master/docs/pretrained_models.md に公開されている，50 ヵ国語以上の言語に対応した，768 次元の単語ベクトルを生成するマルチリンガルモデルを用いた．これはDisilBERT[11] と同様の手法によりモデルの軽量化を図った DistilRoBERTa と呼ばれるモデルである．

2.3.6 ALBERT

ALBERT とは BERT の派生モデルで，2019 年 9 月 26 日に Google から発表された[10]．ALBERT では BERT を軽量化し，かつ，学習を高速化させるため，因数分解埋め込みパラメータとクロスレイヤーパラメータ共有という概念が取り入れられており，いくつかのタスクで BERT の性能を上回り SOTA を打ち立てた．

本研究では https://github.com/alinear-corp/albert-japanese に公開されている，日本語版 Wikipedia のコーパスを学習し，768 次元の単語ベクトルを生成するモデルを用いた．

2.3.7 DistilBERT

DistilBERT とは BERT の派生モデルで，2019 年 10 月 2 日に Hugging Face から発表された[11]．DistilBERT はパラメータを圧縮し軽量化するため，知識蒸留 (Distillation) を用いて学習する．その結果，BERT の約 97% の精度を保ちながら約 60% の高速化を実現させた．

本研究では https://github.com/UKPLab/sentence-transformers/blob/master/docs/pretrained_models.md に公開されている，50 ヵ国語以上の言語に対応した，768 次元の単語ベクトルを生成するマルチリンガルモデルを用いた．

2.3.8 ELECTRA

ELECTRA とは BERT の派生モデルで，2020 年 3 月 23 日に Google から発表された[12]．

BERT 系の自然言語処理モデルは，事前学習に MLM (Masked Language Model) という手法を用いている (Fig. 10).

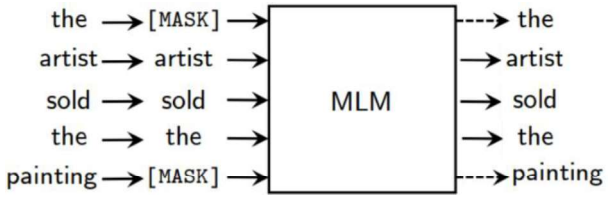

Fig. 10: Mechanism of MLM.

ELECTRA はこの MLM の入力文章全体のうち 15% のみしか学習できないという問題点を解決するため，Fig. 11 のように大小 2 つの BERT を用い

て学習しており，Generator では入力文章に似た文章を生成し，Discriminator は正しい文章かどうかを判別している．

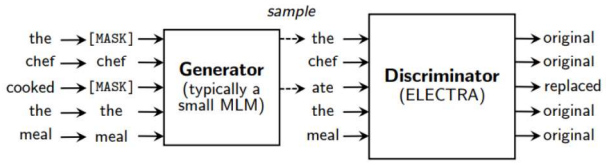

Fig. 11: Scheme of ELECTRA[12].

この学習法により，ELECTRA は RoBERTa の約 1/4 の学習量で RoBERTa と同等の性能を発揮できることが知られている．

本研究では https://huggingface.co/Cinnamon/electra-small-japanese-discriminator に公開されている，256 次元の単語ベクトルを生成するモデルを用いた．

2.4 ベクトル間の距離

本研究では自然言語処理におけるクラスタリングによく用いられているCos 類似度，ユークリッド距離，マンハッタン距離，チェビシェフ距離，マハラノビス距離の 5 つを文章類似度の計算に使用した．

Fig. 12 はベクトル V_A とベクトル V_B の Cos 類似度，ユークリッド距離，マンハッタン距離，チェビシェフ距離を 2 次元で図示したもの，Fig. 13 はマハラノビス距離のイメージを 2 次元で図示したものである．

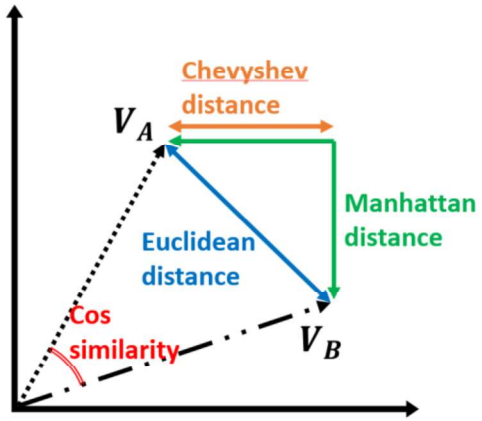

Fig. 12: Scheme of Cos Similarity, Euclidean Distance, Manhattan Distance and Chevyshev Distance.

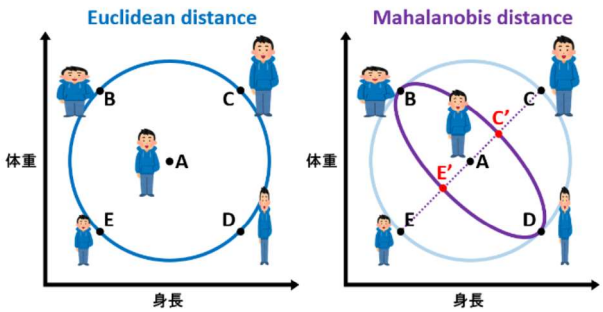

Fig. 13: Scheme of Mahalanobis Distance.

2.4.1 Cos 類似度

Cos 類似度とは，ベクトル同士のなす角度の近さを表しており，ベクトル V_A とベクトル V_B の類似度 S は式 (1) のようにあらわされる．1 に近ければ近いほど，そのベクトル同士が類似していることを意味する．

$$S = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2}\sqrt{\sum_{i=1}^{n} {B_i}^2}} \tag{1}$$

2.4.2 ユークリッド距離

ユークリッド距離とは，2 点間の直線距離であり，ベクトル V_A とベクトル V_B の距離 D_{Euc} は式 (2) のようにあらわされる．

$$D_{Euc} = \sqrt{\sum_{i=1}^{n} (A_i - B_i)^2} \tag{2}$$

2.4.3 マンハッタン距離

マンハッタン距離は，各座標の差の総和を 2 点間の距離とし，ベクトル V_A とベクトル V_B の距離 D_{Man} は式 (3) のようにあらわされる．

$$D_{Man} = \sum_{i=1}^{n} |A_i - B_i| \tag{3}$$

2.4.4 チェビシェフ距離

チェビシェフ距離は，各座標の差の最大値を 2 点間の距離とし，ベクトル V_A とベクトル V_B の距離 D_{Che} は式 (4) のようにあらわされる．

$$D_{Che} = \max(|A_i - B_i|) \tag{4}$$

2.4.5 マハラノビス距離

マハラノビス距離は，相関が強い方向の距離を実際の距離よりも相対的に短く計算する．例えば Fig. 12 のように身長と体重をプロットした平面を考える．ここで B, C, D, E の体形がどれだけ A と似ているかというと，C と E が比較的似ているのに対し，B と D が比較的異なっていることがわかる．しかしユークリッド距離だとA と B, C, D, E の距離は全て等しくなっている．

これを，原点と A を通る直線に平行な相関が強いため，相対的にこの向きの距離を短く計算する（マハラノビス距離）．そうすると A と C', E' の距離が，A と B, D の距離に比べ小さくなり，A とそれぞれの距離の妥当性が増す．

ベクトル V_A とベクトル V_B のマハラノビス距離 D_{Mah} は式 (5) のようにあらわされる．ただし，Σは分散共分散行列である．

$$D_{Mah} = \sqrt{(A_i - B_i)^T \sum^{-1} (A_i - B_i)} \tag{5}$$

3　ユーザー同士の文章類似度とそのフォロー状況の関係の分析

この実験では，ユーザー同士の文章類似度とそのフォロー状況の相関を分析することにより，DVM に適した文章のベクトル化方法とベクトル間の距離算出方法の組み合わせの検討を行った．

文章のベクトル化は Word2Vec, fastText, Doc2Vec, BERT, DistilRoBERTa, ALBERT, DistilBERT, ELECTRA を用いて行った．これらは大きく Word2Vec とその派生形 (fastText, Doc2Vec)，BERT とその派生形 (M-BERT, M-DistilRoBERTa, ALBERT, M-DistilBERT, ELECTRA) に分けることができる．

Word2Vec とその派生形 (fastText, Doc2Vec) に関して，Word2Vec と fastTextについては，モデルを用いて文章中に含まれる全単語ベクトルを生成し，その平均値や最大値を蓄積することにより，1 つの文章から 3 種類の文章ベクトルを生成した (Fig. 14)．Doc2Vec については例外で，文章ベクトルをモデルから直接生成した．

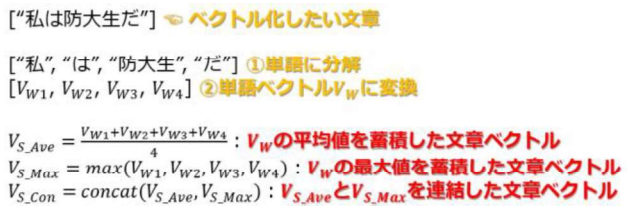

Fig. 14: Sentence vectors created from word vectors.

BERT とその派生形 (M-BERT, M-DistilRoBERTa, ALBERT, M-DistilBERT, ELECTRA) については Sentence Transformer[16] と呼ばれるフレームワークを用いて，文章中に含まれる全単語ベクトルを生成し，その平均値を蓄積することにより，文章ベクトルを生成した．

実験で用いた手法について，手法名，文章ベクトルの次元数，文章ベクトルの作成方法の順で以下に示す．

1) Word2Vec_Ave, 300, 文章に含まれる単語を Word2Vec によりベクトル化し，各次元の平均値を蓄積
2) Word2Vec_Max, 300, 文章に含まれる単語を Word2Vec によりベクトル化し，各次元の最大値を蓄積
3) Word2Vec_Con, 600, Word2Vec_Ave と Word2Vec_Max で生成されたベクトルを連結
4) fastText_Ave, 300, 文章に含まれる単語を fastText によりベクトル化し，各次元の平均値を蓄積
5) fastText_Max, 300, 文章に含まれる単語を fastText によりベクトル化し，各次元の平均値を蓄積
6) fastText_Con, 600, fastText _Ave と fastText _Max で生成されたベクトルを連結
7) Doc2Vec, 300, 文章を Doc2Vec によりベクトル化
8) BERT, 768, 文章を Sentence Transformer と BERT によりベクトル化
9) M-BERT, 768, 文章を Sentence Transformer と BERT multi lingual version によりベクトル化
10) M-DistilRoBERTa, 768, 文章を Sentence Transformer と DistilRoBERTa multi lingual version によりベクトル化
11) ALBERT, 768, 文章を Sentence Transformer と ALBERT によりベクトル化
12) M-DistilBERT, 768, 文章を Sentence Transformer と DistilBERT multi lingual version によりベクトル化
13) ELECTRA, 256, 文章を Sentence Transformer と ELECTRA によりベクトル化

ベクトル間の距離算出方法としては，Cos 類似度，ユークリッド距離，チェビシェフ距離，マンハッタン距離，マハラノビス距離を用いた．

3.1　方法

まず，Twitter からランダムにユーザーを抽出し，約 200 万人分，ユーザー 1 人あたり 100 ツイートのデータを収集し，これを Base 群と定義した．

フォロー関係のないユーザーペア群を Stranger 群，片方向にフォロー関係のあるユーザーペア群を Fan 群，双方向にフォロー関係のあるユーザーペア群を Friend 群と定義し，Base 群と同様，各群ユーザー 1 人あたり 100 ツイートのデータを時間的に可能な限り数多く収集した．Stranger 群は Base 群から互いにフォロー関係にないペアをランダムに抽出，Fan 群と Friend 群は ランダムに取り出した Base 群ユーザーのフォローリストからフォロイー（ユーザーが一方的にフォローしている相手）もしくは相互フォロワーを抽出することにより作成し，各群約 10 万ペアを収集した．

そして，各ベクトル化手法，ベクトル間の距離ごとに各群のペアの文章類似度を計算し，ヒストグラムにより分布を可視化し，Strange, Fan, Friend がどれほど分離できているかにより手法の性能を評価した．

3.2　結果

Word2Vec とその派生形 (Word2Vec, fastText, Doc2Vec) を用いた場合は総じて Stranger, Fan, Friend の分離の性能が高かった．各手法で計算した文章類似度の上位 1, 3, 5, 10, 20, 30% における Stranger, Fan, Friend の占める割合を示した図 (Fig. 18-23) をみれば，Word2Vec とその派生形を用いた場合，BERT とその派生形 (BERT, M-BERT, M-DistilRoBERTa, ALBERT, M-DistilBERT, ELECTRA) を用いた場合に比べて，文章類似度の上位に含まれる friend の割合が高いことがわかる．

最も良く Stranger, Fan, Friend を分離できたのは Doc2Vec と Cos 類似度の組み合わせ (Fig. 15) であり，文章類似度の上位 1, 3, 5, 10, 20, 30% 全てにおいて，Friend の占める割合が他の組み合わせに比して大きい．

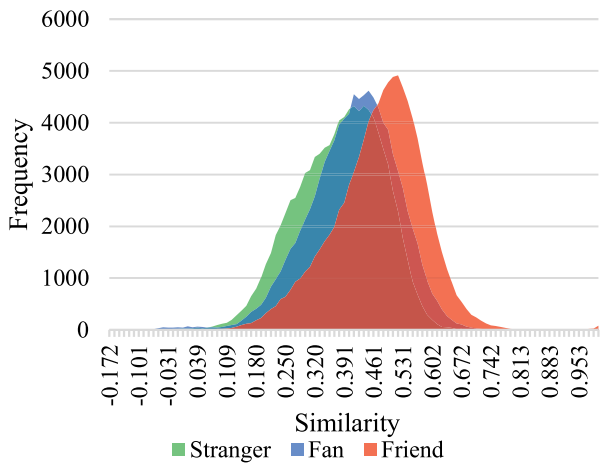

Fig. 15: Frequency distribution of Cos Similarity between user vectors calculated by Doc2Vec.

BERT とその派生形の中で最も良く Stranger, Fan, Friend を分離できたのは ALBERT と Cos 類似度の組み合わせ (Fig. 16) である.

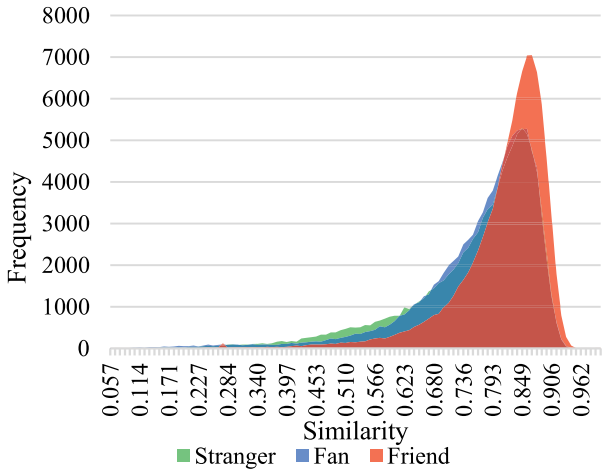

Fig. 16: Frequency distribution of Cos Similarity between user vectors calculated by ALBERT.

マハラノビス距離を用いた場合は総じて分離の性能が低く，文章類似度の取りうる値が1点に集中している．例として Doc2Vec と マハラノビス距離の組み合わせで分離した文章類似度の分布図を Fig. 17 に示す.

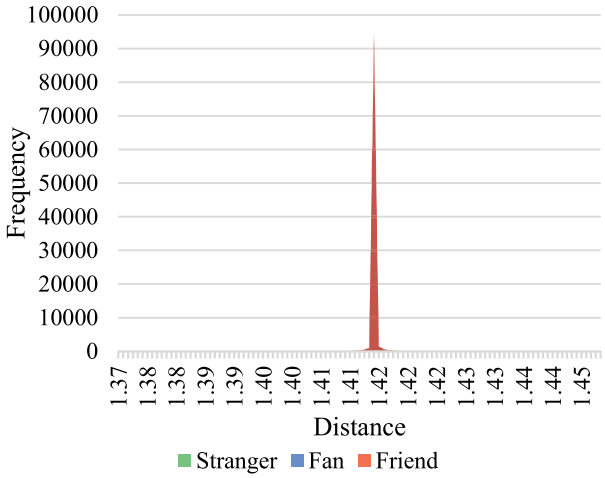

Fig. 17: Frequency distribution of Mahalanobis distance between user vectors calculated by Doc2Vec.

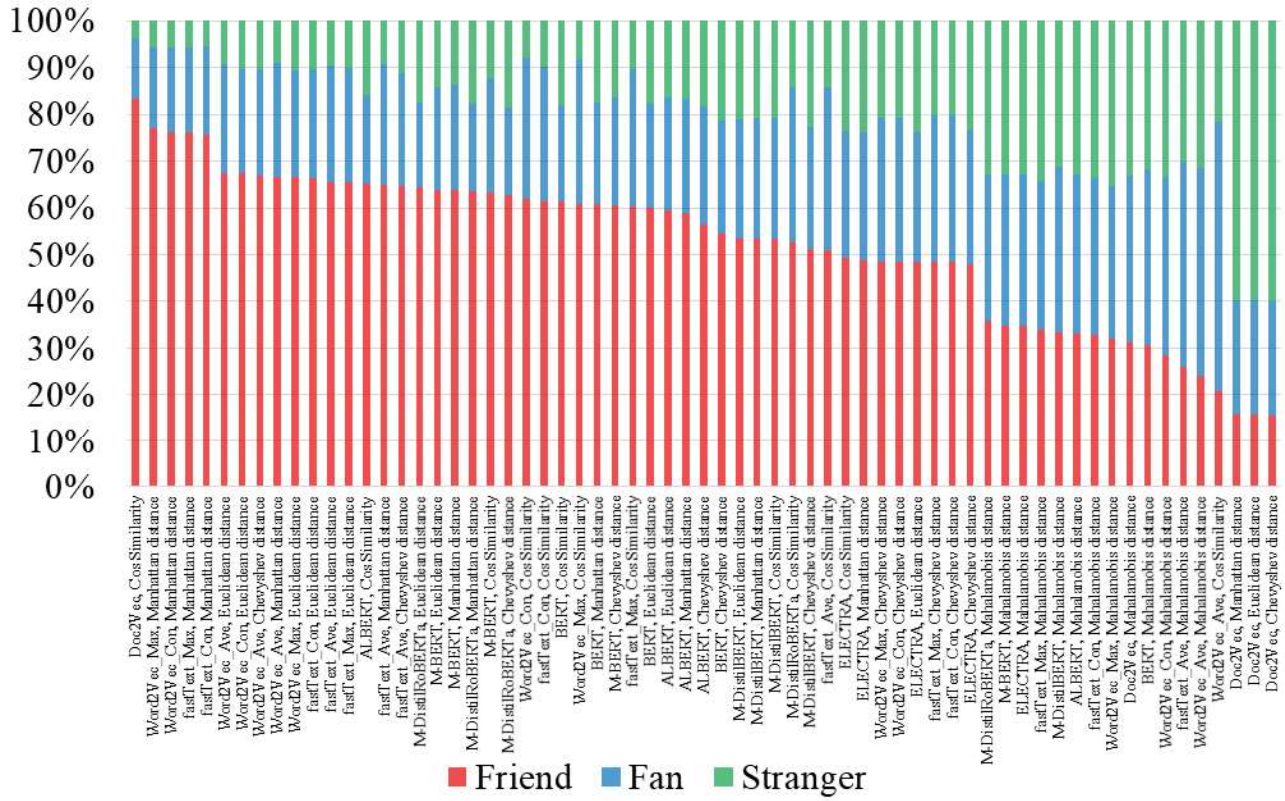

Fig. 18: Ratio of Stranger, Fan and Friend in the top 1% of the document vector similarity.

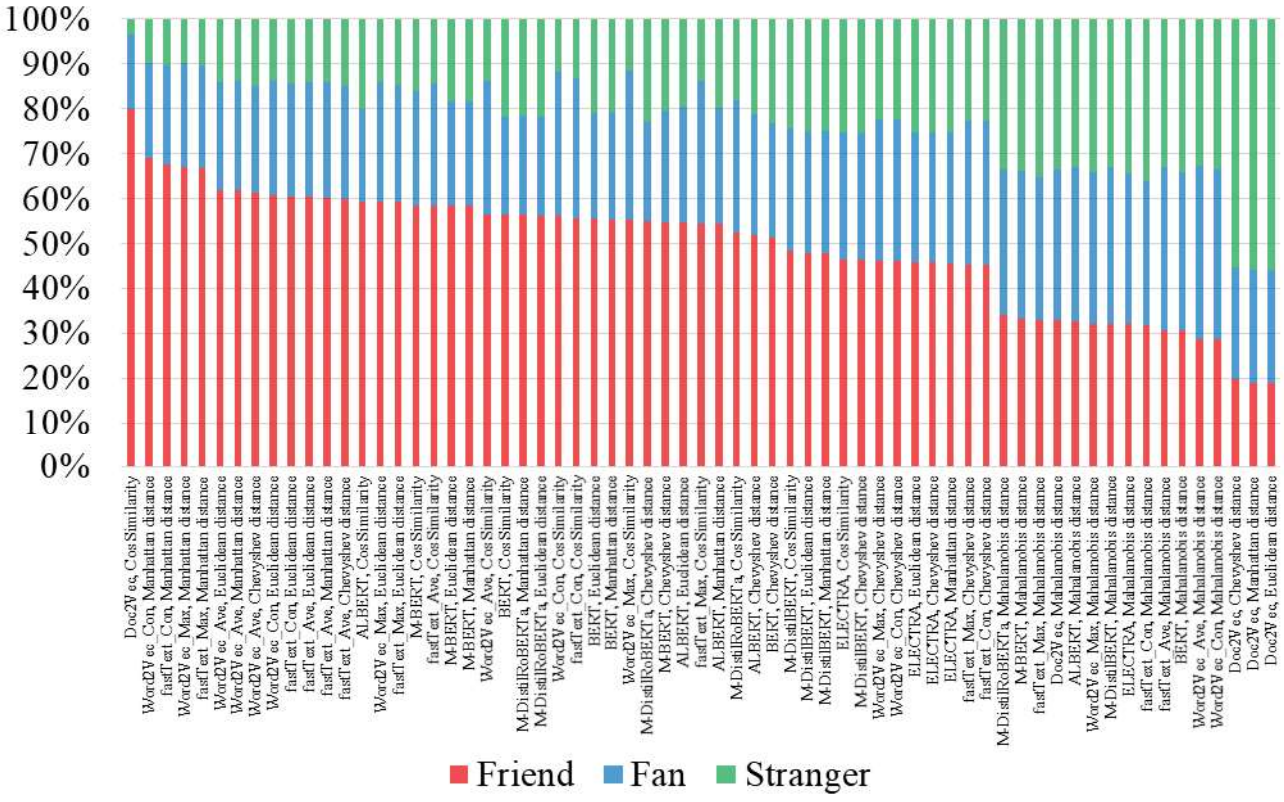

Fig. 19: Ratio of Stranger, Fan and Friend in the top 3% of the document vector similarity.

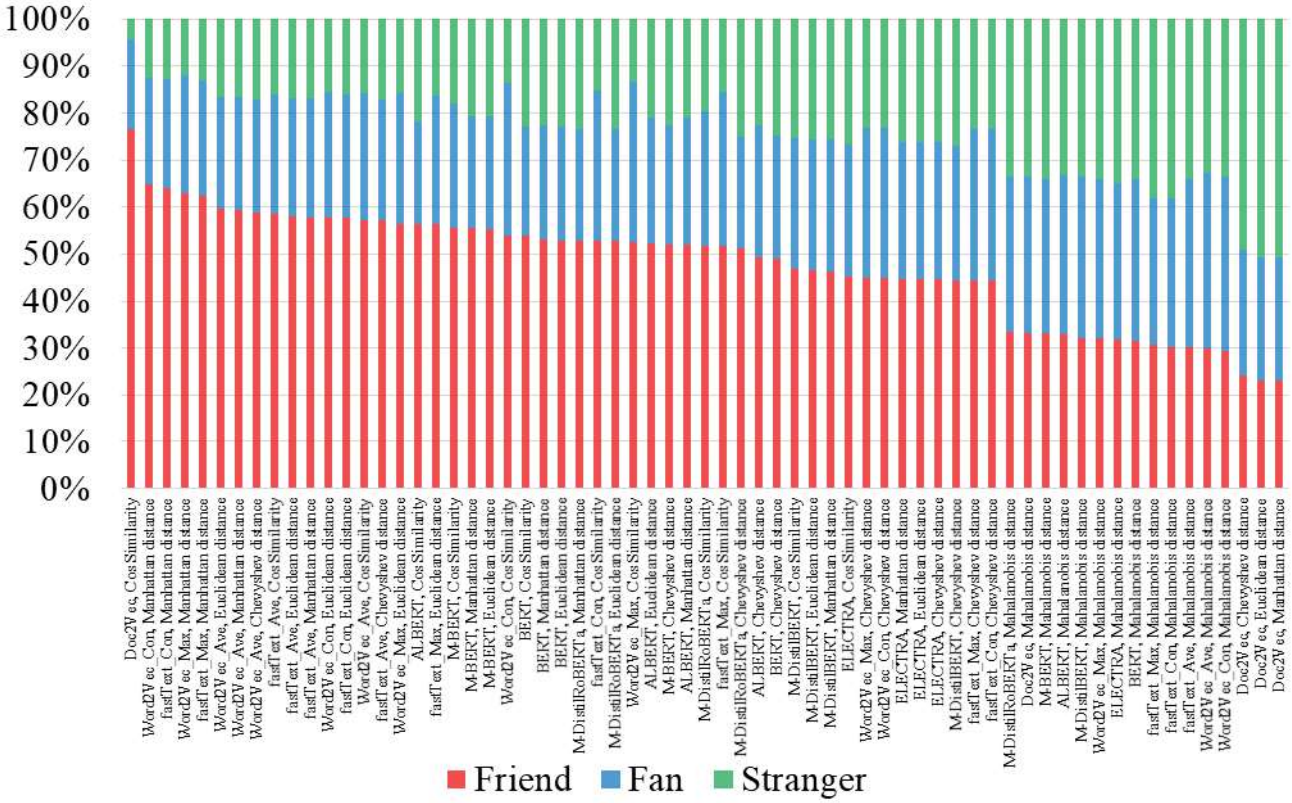

Fig. 20: Ratio of Stranger, Fan and Friend in the top 5% of the document vector similarity.

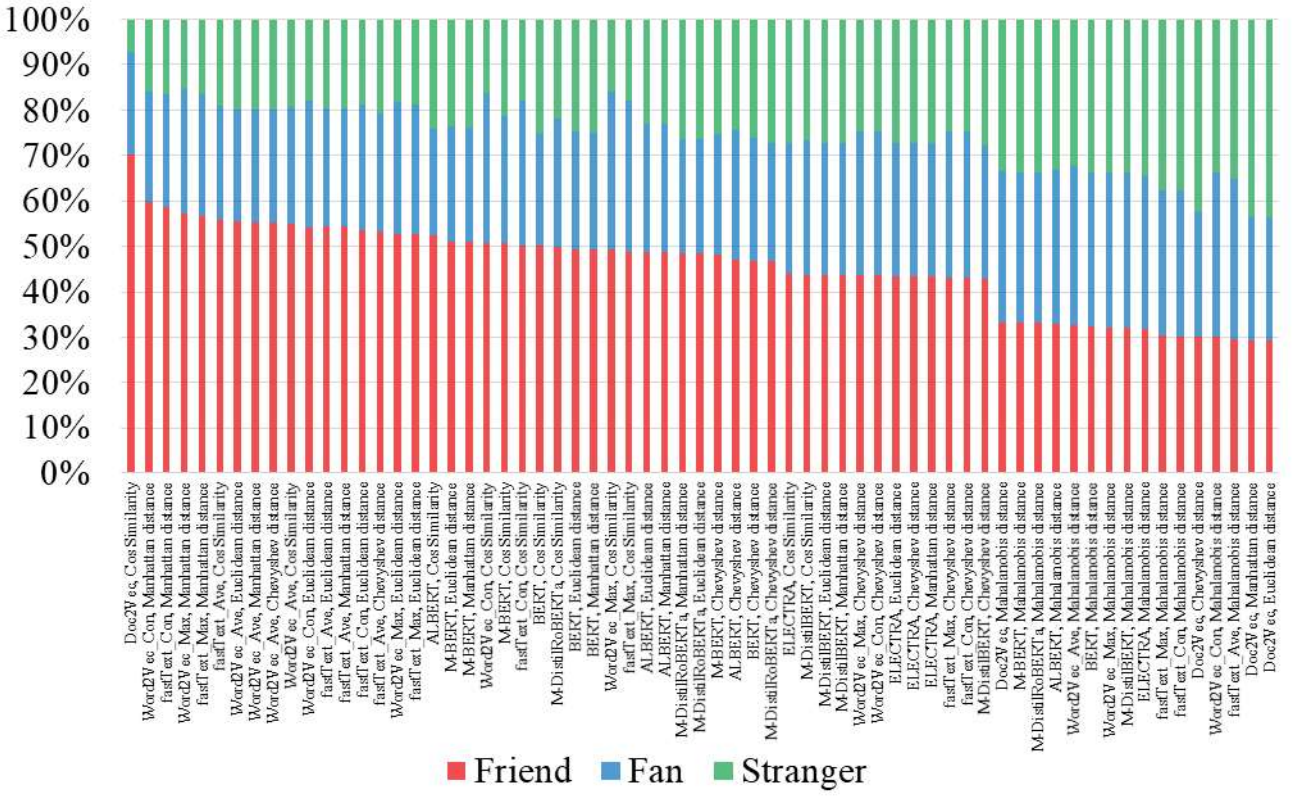

Fig. 21: Ratio of Stranger, Fan and Friend in the top 10% of the document vector similarity.

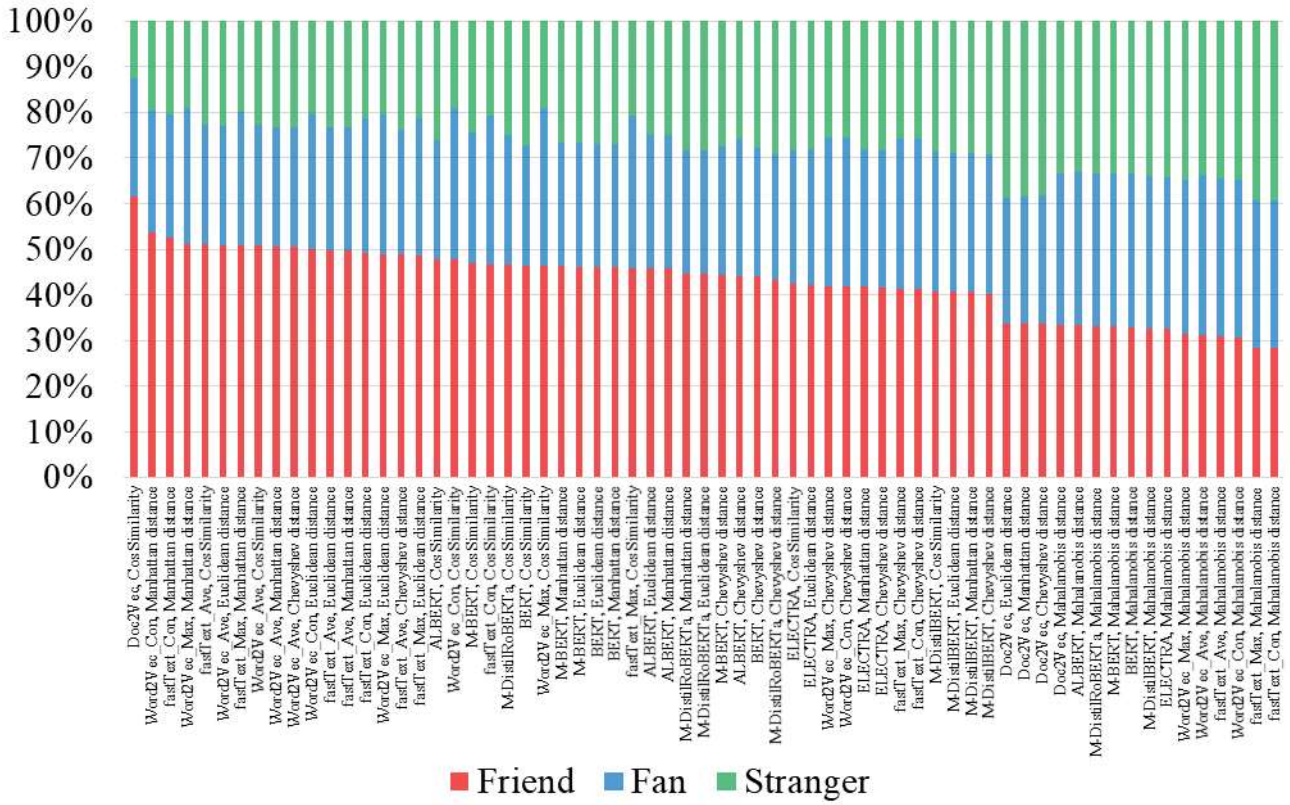

Fig. 22: Ratio of Stranger, Fan and Friend in the top 20% of the document vector similarity.

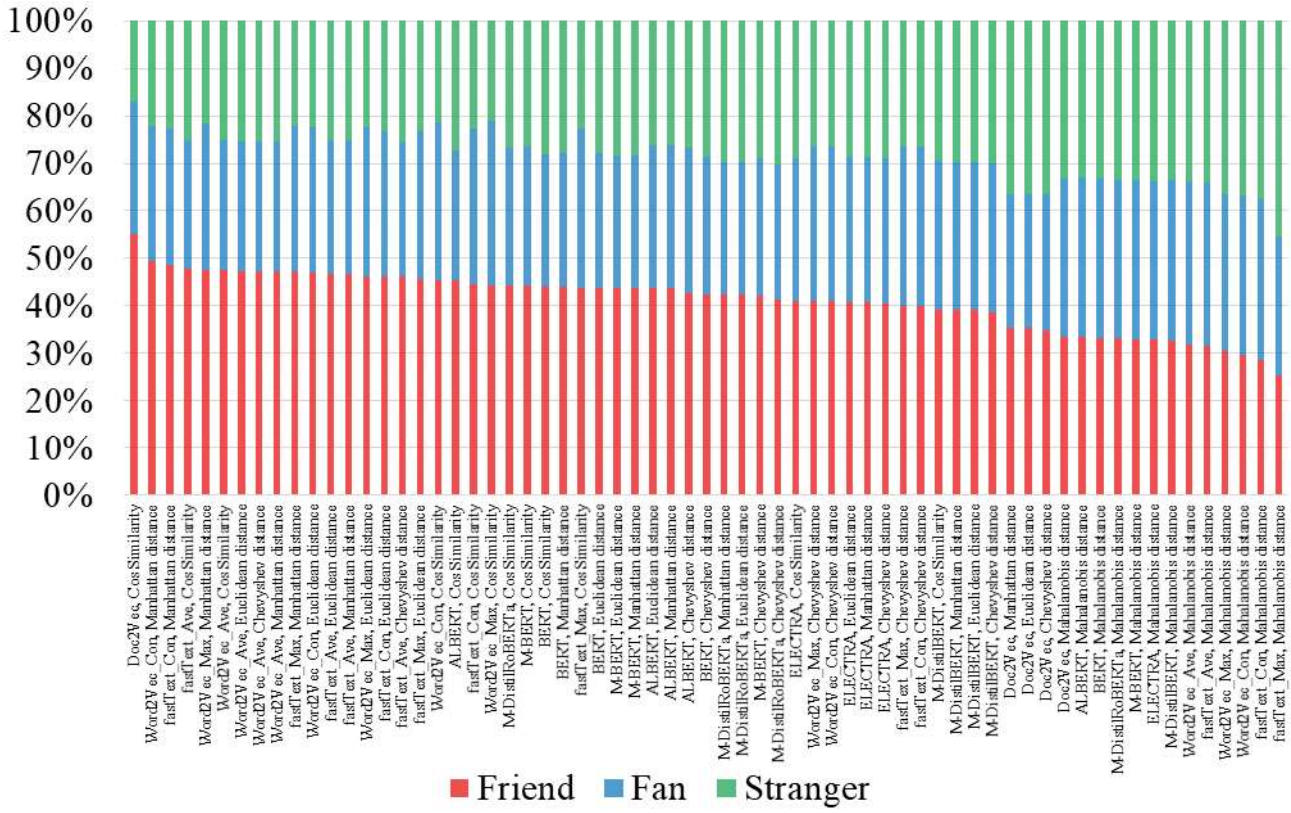

Fig. 23: Ratio of Stranger, Fan and Friend in the top 30% of the document vector similarity.

3.3 考察

Word2Vec とその派生形の分離性能が良かったことから，CBow や PV-DM といった文章の学習モデルが日本語の文章の特徴量抽出に向いていることが考えられる．

また，実験で使用した全モデルの中で，文章の特徴量を学習した唯一のモデルである Doc2Vec が最も良い結果を示したことから，単語ベクトルの蓄積から生成された文章ベクトルよりも，文章の特徴量を学習したモデルから直接生成した文章ベクトルを文章の比較に用いたほうが良いと考えられる．

ベクトル間の距離算出方法に関して，Fig. 15, 16 にある通り，性能が良かった組み合わせには Cos 類似度が用いられていたが，必ずしも Cos 類似度が最も文章類似度を計算する上で適しているわけではないといえる．Fig. 18-23 の横軸にある組み合わせは，左側であればあるほどフォロー関係を文章類似度により分離できており，性能が高いことを示しているが，例えば Fig. 18 で最も性能の高い組み合わせである Doc2Vec, Cos similarity に引き続いて性能の高かったのは Word2Vec_Max, Manhattan distance の組み合わせである．それぞれの自然言語処理モデルには，計算上の特性がある．そのため，それぞれのモデルや手法に合った距離の計算方法があると考えるのが妥当である．

今回用いたモデルは全て日本語版 Wikipedia のコーパスを学習したものであるが，Twitter のデータを学習させることにより更に性能が向上する可能性がある．

4 Doc2Vec と Cos 類似度による DVM の有効性の検証

この実験では，DVM を用いて Twitter のユーザーにリコメンドを行い，どれほどエンゲージメントが発生するかを調べることにより，DVM の有効性を検証した．

実験に用いた DVM には，ユーザー同士の文章類似度とそのフォロー状況の相関の分析で，最も良くフォロー関係を分離した Doc2Vec と Cos 類似度の組み合わせを実装した．

4.1 方法

まず，Twitter アカウント @nocodehakase の最新 100 ツイートを Doc2Vec を使用しベクトル化した（ここで用いるアカウントは任意のもので良く，@nocodehakase に協力してもらった）．そして，ランダムに収集したツイッターユーザー約 14 万人に関しても同様にツイートをベクトル化し，@nocodehakase のベクトルとの Cos 類似度が高い順に，上位 0.1%, 0.3%, 1%, 3%, 10%（以上，実験群），11-100%（対照群）のグループに分けた．

次に，各グループからランダムに抽出した 90 ユーザーに対し，@nocodehakase をリコメンドすることに相当するアクションとして，@nocodehakase から各ユーザーへの最新ツイートのいいねを行った．

@nocodehakase のアクション実施から 72 時間後，いいねを付けたユーザーからのエンゲージメント（いいね，リツイート，リプライ，フォローバック）を集計した．

4.2 結果

上位 1% もしくはそれ以上の実験群で，対象群に比べ，高いエンゲージメント数，エンゲージメント率が得られた (Table 1, Fig. 24).

Table 1: Engagement ratio of each group.

	ｲｲﾈ	ﾘﾂｲｰﾄ	ﾘﾌﾟﾗｲ	ﾌｫﾛﾊﾞ	計	ｴﾝｹﾞｰｼﾞﾒﾝﾄ率
0.1%	1	0	0	3	4	4.4%
0.3%	3	0	0	4	7	7.8%
1%	0	0	0	4	4	4.4%
3%	0	0	0	1	1	1.1%
10%	1	0	0	2	3	3.3%
対照	0	0	0	1	1	1.1%

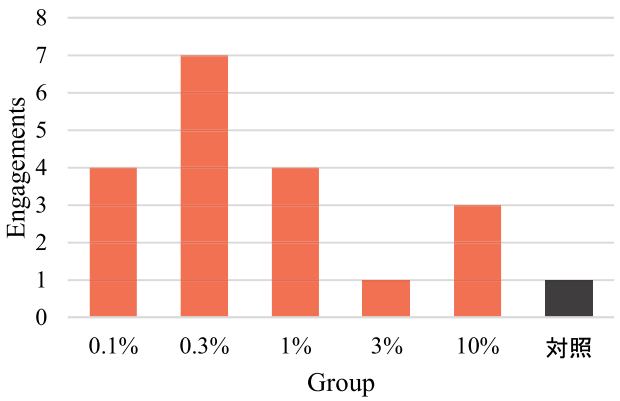

Fig. 24: Engagements of each group.

上位 1% 以上のグループで平均 5.53% のエンゲージメント率が得られたが，この数字は主要 SNS の既存リコメンドシステムにおけるエンゲージメント率に比して非常に高い (Fig. 25).

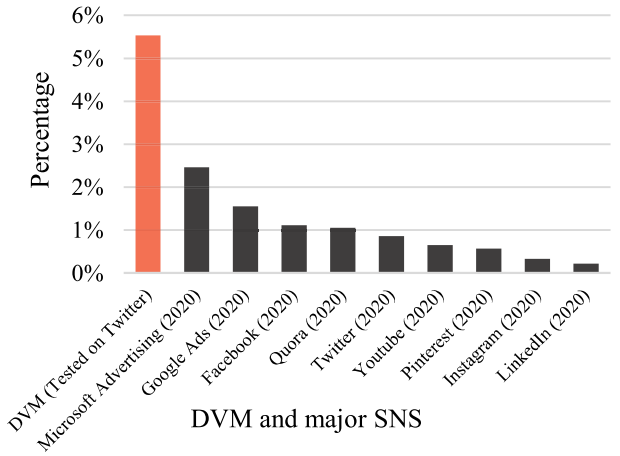

Fig. 25: Engagement ratio of each SNS[17] and DVM.

4.3 考察

DVM を用いてのリコメンドで既存のリコメンドシステムに比べ，高いエンゲージメント率が得られたことから，文章類似度を基にリコメンドの対象を絞ることにより，無差別に対象を絞るときよりも，多くのエンゲージメントが期待できるといえる．

しかし，まだ 1 つのアカウントを用いてしか実験をすることができておらず，様々な嗜好の Twitter アカウントを用いて DVM の実験を行わなければ，普遍的な有効性は認められないといえる．

5 まとめ

ユーザー同士の文章類似度に基づき，フォロー関係が成立しているかどうかを一定以上の水準で分離できたことから，文書類似度によりユーザー間でフォローが成立し得るかどうかを弁別できるということが示唆された．また，本研究で構築したユーザーリコメンドシステムが既存の SNS におけるリコメンドシステムに比して非常に高い性能を示したことから，文書類似度を基準とすることにより，精度の高いリコメンドが可能になるということが明らかとなった．つまりDVM の高い有効性が示された．

本研究で構築した DVM は，単体でも十分に効果を発揮するが，既存のリコメンドシステムに組み込むことにより，ユーザー同士のリコメンドに加え，ユーザーに紐づけられた購買・閲覧履歴等のデータ同士をリンクさせ，リコメンドをするといった活用方法が考えられる．

自然言語処理モデルは今なお研究が活発に続いている分野であり，今回用いたモデル以外にも日本語の文章の比較に適したモデルがある，もしくはこれから出てくる可能性がある．そのため，本研究で扱ったテーマは継続して研究していく余地があるといえる．

参考文献

1) 北　栄輔, リコメンド・サービス・コンテスト実施報告, 情報処理 Vol. 50 No.4, pp. 334-341, 2009.
2) 神嶌 敏弘, 推薦システムのアルゴリズム, 人工知能学会誌 Vol. 23 No. 1, pp. 89-103, 2008.
3) 渡邉裕介; 佐藤浩; 白川智弘, 文章類似度に基づくユーザーリコメンドシステムの提案, 計測自動制御学会システム・情報部門　第 62 回システム工学部会研究会論文集, pp. 8-16, 2020.
4) Harris Zellig, Distributional structure, Word Vol. 10 No. 2-3, pp. 146-162, 1954.
5) Quoe Le; Tomas Mikolov, Distributed Representations of Sentences and Documents, Proceedings of the 31st International Conference on International Conference on Machine Learning Vol. 32, pp. 1188-1196, 2014.
6) Jacob Devlin; Ming-Wei Chang; Kenton Lee; Kristina Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies Vol. 1, pp. 4171-4186, 2019.
7) Tomas Mikolov; Kai Chen; Greg Corrado; Jeffrey Dean, Efficient Estimation of Word Representations in Vector Space, CoRR abs/1301.3781, 2013.
8) Piotr Bojanowski; Edouard Grave; Armand Joulin; Tomas Mikolov, Enriching Word Vectors with Subword Information, Transactions of the Association for Computational Linguistics Vol. 5, pp. 135-146, 2017.
9) Yinhan Liu; Myle Ott; Naman Goyal; Jingfei Du; Mandar Joshi; Danqi Chen; Omer Levy; Mike Lewis; Luke Zettlemoyer; Veselin Stoyanov, RoBERTa: A Robustly Optimized BERT Pretraining Approach, CoRR abs/1907.11692, 2019.
10) Zhenzhong Lan; Mingda Chen; Sebastian Goodman; Kevin Gimpel; Piyush Sharma; Radu Soricut, ALBERT: A Lite BERT for Self-supervised Learning of Language Representations, CoRR abs/1909.11942, 2019.
11) Victor Sanh; Lysandre Debut; Julien Chaumond; Thomas Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, CoRR abs/ 1910.01108, 2019.
12) Kevin Clark; Minh-Thang Luong; Quoc V. Le; Christopher D. Manning, ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators, CoRR abs/ 2003.10555, 2020.
13) 藤倉まなみ; 大和妃香里; 福岡雅子,「インスタ映え」料理写真の SNS 掲載による食べ残し増加の可能性, 第 29 回廃棄物資源循環学会研究発表会　公演原稿, pp. 105-106, 2018.
14) Jörg Landthaler; Bernhard Waltl; Dominik Huth; Daniel Braun; Christoph Stocker; Thomas Geiger; Florian Matthes, Extending Thesauri Using Word Embeddings and the Intersection Method, Proc. of 2nd Workshop on Automated Semantic Analysis of Information in Legal Texts Vol. 2143, 2017.
15) Ashish Vaswani; Noam Shazeer; Niki Parmar; Jakob Uszkoreit; Llion Jones; Aidan N. Gomez; Lukasz Kaiser; Illia Polosukhin, Attention Is All You Need, Advances in Neural Information Processing Systems, pp. 5998-6008, 2017.
16) Nils Reimers; Iryna Gurevych, Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks, Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pp. 3982-3992, 2019
17) ADSTAGE, Q1 2020 Paid Media Benchmark Report, https://www.adstage.io/resources/q1-2020-ppc-benchmark-report/ (Accessed in Mar. 3, 2021)