Vol.2015-DBS-162 No.11
2015/11/26

Paragrph Vectorへの追加情報の効率的な埋め込み

橋戸 拓也[1,a)]　新妻 弘崇[2,b)]　太田 学[2,c)]

概要：本稿では, Paragraph Vector を拡張した ScoreSent2Vec を提案する．Paragraph Vector とは, 文章の特徴を数百次元程度の固定長の密な数値ベクトルとして表す手法である．ScoreSent2Vec は, ある文章に関わる追加情報が文章以外の形で表されているときに用いる．例えば, Twitter に投稿された文章に画像が添付されている場合などに利用できる．このような場合, ScoreSent2Vec では追加情報を単語の 1 つとみなして Paragraph Vector に入力することで数値ベクトルを得ることができる．ScoreSent2Vec で得たベクトルを用いると，単純に追加情報を組み合わせた場合よりも高い精度で文章の分類ができることを 2 種類の実験で示す．1 つは英文の分類で, 助動詞の出現回数を追加情報として利用した実験を示す．もう 1 つは Twitter に投稿された文章を伴う画像の分類で, 画像の色ヒストグラムを追加情報として利用した実験を示す．どちらの実験でも, Paragraph Vector に単純に追加情報を組み合わせた場合より, ScoreSent2Vec を用いた方が高い分類精度を示した．

Efficient Embedding of Additional Information into Paragraph Vectors

Takuya Hashito[1,a)]　Hirotaka Niitsuma[2,b)]　Manabu Ohta[2,c)]

1. はじめに

従来，文や文章などの特徴を表現するのに，BOW (Bag-of-Words) 等の単語の出現頻度に基づく特徴が広く使われてきた．しかし，単語の出現頻度のみでは語順の違いを考慮できないため，同義語が特徴空間上で近くに出現しないなどの問題があった．

近年，この問題を解決する手法として word2vec[1][2] が注目されている．word2vec は Mikolov らが提案した単語の特徴を低次元の密な数値ベクトルで表現する手法である．ベクトルで表現することで, 単語同士の類似度などを測定できる．このベクトル表現は分布表現やベクトル分散表現とも呼ばれている．word2vec は文章中の単語を予測する擬似的なタスクをニューラルネットワークで学習することで，そのニューラルネットワークの中間層の値を，単語のベクトル分散表現として抽出する手法である．単語を予測する擬似的なタスクとしては，中心の単語から周辺の単語を予測するタスクと, 周辺の単語から中心の単語を予測するタスクがあり，それぞれ Skip-gram モデルと CBOW (Continuous Bag-of-Words) モデルと呼ばれている [1][2].

word2vec は単語の意味を適切に表現した数値ベクトルの計算は可能であるが，文や文章の意味を適切に表現した数値ベクトルを計算することはできない．そこで word2vec の Skip-gram と CBOW のモデルに文章全体を数値ベクトルで表現できるニューラルネットワークを追加する拡張をしたのが Paragraph Vector[3] である．Skip-gram モデルを拡張したものは PV-DBOW (Paragraph Vector with Distributed Bag of Words)，CBOW モデルを拡張したものは PV-DM (Paragraph Vector with Distributed Memory) と呼ばれている．映画のレビューをした英文から，そのレビュアがつけた映画の評価値を文章の特徴のみから推定する問題 [4] において，Paragraph Vector は従来の BOW などを使った手法よりも高い精度を実現できることが報告されている [3].

本研究では Paragraph Vector を拡張することで, 文章に追加情報が伴っている場合に，その Paragraph Vector に

[1] 岡山大学工学部情報系学科
[2] 岡山大学大学院自然科学研究科
[a)] pldu28hq@s.okayama-u.ac.jp
[b)] niitsuma@suri.cs.okayama-u.ac.jp
[c)] ohta@de.cs.okayama-u.ac.jp

適切に追加情報を埋め込んだ特徴ベクトルを計算することで分類精度を高めることを目的とする．具体的には文章に画像などの追加情報が伴っている場合に，その追加情報を埋め込んだ低次元の特徴ベクトルを使って分類精度を高める方法について議論する．一般的に，文章には様々な追加情報が伴っていることが多い．例えば，Amazon.com[*1] や価格.com[*2] などの商品のユーザレビューには，その商品の評価スコアがレビューの文章と共に掲載されている．また，画像や音声などの複雑な情報を伴う文章も多く存在する．本研究では，これらの追加情報を比較的低次元の数値ベクトルとして表現する場合に注目する．例えば文章に画像が伴っているならば，その画像を減色した色ヒストグラムなどを使えば，追加情報を低次元の数値ベクトルで表現できる．商品の値段を伴った商品の説明文章ならば，値段を 1 次元のベクトルで表すことができる．このような文章に付随する情報を適当な数値ベクトルで表現できる場合，そのベクトルを以下では score vector と呼ぶこととする．score vector を文章に含まれる単語の 1 つとして考えると，Paragraph Vector のモデルに score vector を表現するニューラルネットワークを追加することで自然に拡張できる．この拡張により, score vector の情報を Paragraph Vector に埋め込む手法である ScoreSent2Vec を提案する．

2. 関連研究

2.1 word2vec

ここでは従来手法である word2vec のモデルについて説明する．Mikolov らは word2vec を実現するニューラルネットワークの構造として Skip-gram モデルと CBOW モデルを提案している．以下では skip-gram モデルと CBOW モデルについて説明する．

Skip-gram モデル: 図 1 に Skip-gram モデルのニューラルネットワークを示す．このモデルは入力層, 中間層, 出力層からなり, 文章中のある単語 $w(t)$ を入力とし, その前後の単語 $w(t-c), ..., w(t-1), w(t+1), ..., w(t+c)$ を出力とするニューラルネットワークである．c は文脈サイズと呼ばれるパラメータであり，同じ文脈として考慮する前後の単語数である．

中間層 $\boldsymbol{h}$ は以下の式で表される．

$$\boldsymbol{h} = \boldsymbol{w}(t)^{\mathrm{T}}\boldsymbol{W}$$

$\boldsymbol{w}(t)$ は単語 $w(t)$ をハフマンコーディングを使って表現した列ベクトルである．$\boldsymbol{W}$ は入力層から中間層への重み行列である．この行列 $\boldsymbol{W}$ を構成する列ベクトルが単語の意味を表す特徴ベクトルとして word2vec が出力するベクトルとなる．

[*1] http://www.amazon.com
[*2] http://kakaku.com

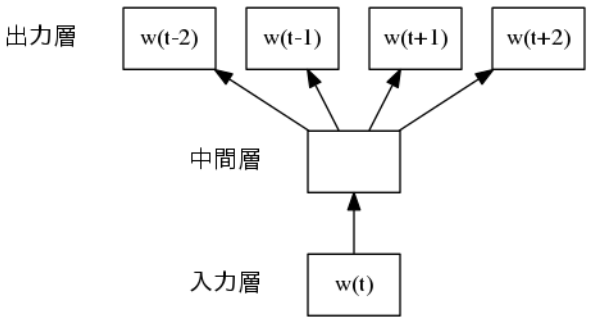


図 1 Skip-gram モデル

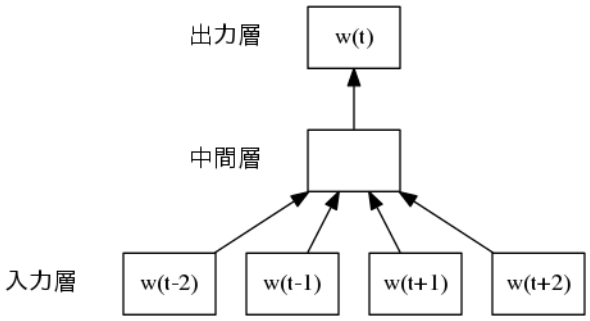


図 2 CBOW モデル

$w(t)$ の前後の単語 $w(t \pm c')$ を表す出力層の値は以下のロジスティック関数を使って求める．

$$\boldsymbol{w}(t \pm c') = \boldsymbol{W'}(\frac{1}{1+\exp(\boldsymbol{h}^{\mathrm{T}})})$$

ここで $\boldsymbol{W'}$ は中間層から出力層への重み行列である．このニューラルネットワークの重み行列 $\boldsymbol{W}$ と $\boldsymbol{W'}$ を学習データとして与えるられる文章，すなわち単語の列から学習することによって，適切な単語の特徴ベクトルを獲得できるのが word2vec である．

CBOW モデル: 図 2 に CBOW モデルのニューラルネットワークを示す．このモデルは Skip-gram モデルと同様に入力層, 中間層, 出力層からなるが，入力と出力が Skip-gram モデルとは逆となる．出力は中心の単語 $w(t)$ であり，入力はその前後の単語 $w(t-c), ..., w(t-1), w(t+1), ..., w(t+c)$ とするニューラルネットワークである．つまり, Skip-gram モデルとは反対に，周辺の単語から中心にある単語を推定する問題をニューラルネットワークに学習させる．

中間層 $\boldsymbol{h}$ は以下の式で表される．

$$\boldsymbol{h} = \frac{1}{c}\left(\sum_{c'=-c,-c+1,...c-1,c} \boldsymbol{w}(t+c')\right)^{\mathrm{T}} \boldsymbol{W}$$

Skip-gram との違いは中間層 h の違いのみで中間層から出力層への重みなどは同じ式で表される．

CBOW モデルでも Skip-gram モデルでも, 入力層と中

間層をつなぐ重み行列が, word2vec が最終的に生成する数値ベクトルとなる.

word2vec を用いた研究は多数報告されている. 岩井ら [5] は word2vec の手法に基づき, 述語項構造に対するベクトル表現を獲得する手法を提案した. 述語項表現とは, ある文中で述語が他の単語とどのような関係にあるのかを記述した構造である. 述語の意味が明示的には表れていない動詞と目的語のペアと，その本来の意味を表す基本動詞との類似度を計測した. 例えば, 動詞と目的語のペア”do-dish”の本来の意味を表す基本動詞は”wash”になる. このような動詞と目的語のペア 10 組と英語基本動詞活用辞典 [6] に収録されている 385 個の基本動詞の類似度を計測した. その結果, 3 組の動詞と目的語のペアと基本動詞については最も高い類似度となり，この 3 組を含む 6 組は上位 10 番以内の類似度となった.

黒崎ら [7] は文章中に含まれる顔文字の感情分類を行うために word2vec を利用した. 具体的には, Twitter に投稿された文章を対象とし, 顔文字と感情を表す語（喜び, 悲しみ, 落胆など 6 種類の感情）との類似度を測り, 最も近い感情をその顔文字が表す感情として, 顔文字を紹介しているウェブサイト上の分類と比較した. 再現率, 適合率, F 値を算出した. F-measure を算出したところ，最も高かったのは“ 喜び ”の 0.8465 で, 最も低かったのは“ 怒り ”の 0.0513 となり, 6 種類の感情全体での F-measure は 0.7457 となった.

2.2 Paragraph Vector

ここでは, word2vec の拡張である Paragraph Vector のモデルについて説明する. Paragraph Vector とは, word2vec と同様の手段で, 単語ではなく文章のベクトル表現を獲得する手法である. 以下では Paragraph Vector の PV-DBOW モデルと PV-DM モデルについて説明する.

PV-DBOW モデル: 図 3 に PV-DBOW モデルのニューラルネットワークを示す. このモデルは word2vec の Skip-gram モデルを拡張したものである. 最初に word2vec の Skip-gram モデルを使って学習したニューラルネットワークを作成する. 次に Skip-gram モデルのニューラルネットワークの入力層を，文章の ID を入力するネットワークと入れ替える. この入れ替えられた層の値のみを word2vec と同様の方法で目的の文章に対して学習することで，文章のベクトル表現を得ることができる.

PV-DM モデル: word2vec の CBOW モデルを拡張したモデルである. 図 4 にそのニューラルネットワークを示す. PV-DBOW モデルと同様に，最初は word2vec の CBOW モデルを使って学習したニューラルネットワークを作成する. 次に，文章の ID を入力するネットワークを入力層に追加する. この追加されたネットワーク部分のみを word2vec と同様の方法で目的の文章に対して学習することで，文章のベクトル表現を得ることができる.

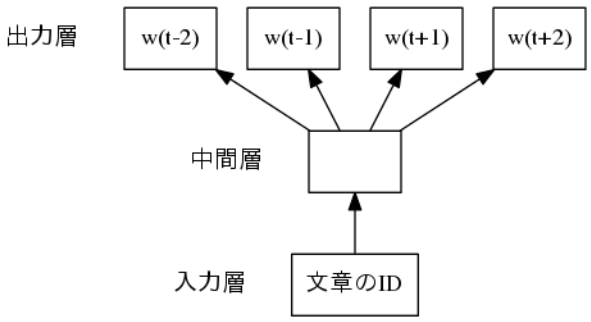


図 3 PV-DBOW モデル

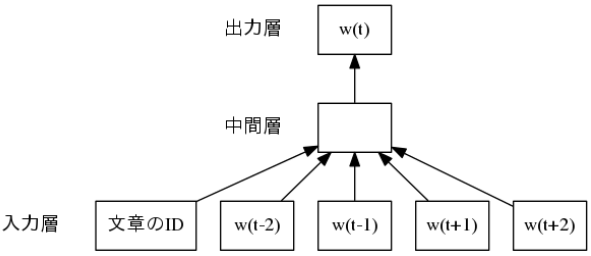


図 4 PV-DM モデル

Paragrh Vector を用いた研究も多数報告されている. 中野ら [8] は提題表現に基づいた重要段落の抽出に Paragraph Vector を用いた. 提題表現とは, 文の主題を取り上げる表現である「〜は」の形式を典型とする. 提案手法では, 文章中に記述している語をベクトル化し, 段落ごとに得たベクトルとの内積を計算することで重要段落を得る. 毎日新聞コーパス（1998-99 年）及び日経新聞コーパス（1998 年）からニュース報道記事に該当するものを選択し, 実験データとしたところ, 毎日新聞記事では 61.2 %, 日経新聞記事では 77.9 %の抽出精度を得た.

佐藤ら [9] はウェブ上の有害な文書を分類するために Paragraph Vector を用いた. 実験では, 有害文書と無害文書をそれぞれ 10 万件ずつ用いて評価実験を行った. 提案手法の中では, PV-DM モデルを拡張した PV-CBOW(Continuous Bag-of-Words of Paragraph Vectors) モデルの F-measuer が最も高く, ベクトルの次元数が 200 の時に 0.9431 であった.

3. 提案手法

文章に付随している画像などの情報を低次元の数値ベクトルで表現できる場合，本研究ではそのベクトルを score

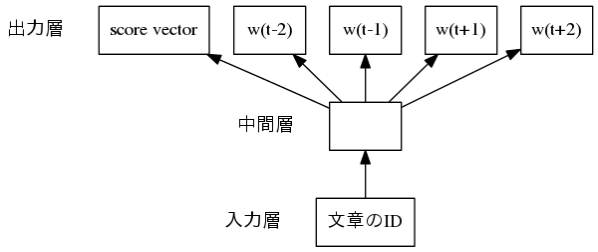


図 5 SPV-DBOW モデル

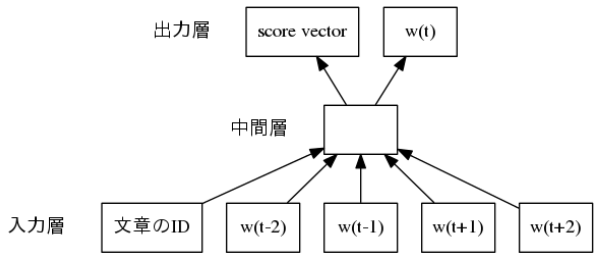


図 6 SPV-DM モデル

vector と呼ぶこととする．score vector を単語の 1 つとして考えると，以下のような Paragraph Vector の拡張を導入することができる．このように拡張した Paragraph Vector のことを，本稿では ScoreSent2Vec と呼ぶ．

本研究では PV-DBOW モデルの拡張として SPV-DBOW (scored Paragraph Vector with distributed bag of words) , PV-DM モデルの拡張として SPV-DM (scored Paragraph Vector with distributed memory) の 2 つのモデルを導入する．以下では SPV-DBOW モデルと SPV-DM モデルについて説明する．

SPV-DBOW モデル: PV-DBOW モデルを拡張し，score vector のベクトル値を予測するニューラルネットワークを追加したモデルである．図 5 にそのニューラルネットワークを示す．Skip-gram モデルと同様に，Paragraph Vector の値から文脈ウインドウに含まれる周辺の単語と score vector の値を同時に予測する問題をニューラルネットワークに学習させることで，適切な Paragraph Vector が学習される．

SPV-DM モデル: PV-DM モデルを拡張し score vector のベクトル値を予測するニューラルネットワークを追加したモデルである．図 6 にそのニューラルネットワークを示す．PV-DM モデルを基礎としており，それ以外は SPV-DBOW モデルと同様の処理を行う．

提案手法のソースコードは次のサイトで公開している
`https://github.com/niitsuma/ScoreSent2Vec`

4. 実験

以下では，提案手法である ScoreSent2Vec のモデルの有効性を実験的に示す．実験は 2 種類で, Brown Corpus[10] に含まれる英文の分類と, Twitter から獲得した柴犬の画像の分類である．Paragraph Vector の実装としては klb3713 らが実装した sentence2vec[*3] を利用した．この実装を以下では Sent2Vec と呼ぶ．

4.1 Brown Corpus の英文の分類

はじめに Brown Corpus に含まれる英文がどのカテゴリに属するかを，文中の助動詞に着目して分類する実験を行う．Brown Corpus は 1961 年にブラウン大学で作成された英文のコーパスで，総語数は約 100 万語となっている．Brown Corpus のテキストは news, religion などの多くのカテゴリに分類されている．このカテゴリの内 news, religion, hobbies, science fiction, romance, humor の合計 6 カテゴリに含まれる 16,964 文を実験に用いる．表 1 に各カテゴリに含まれる助動詞の出現回数を示す．表に示すように助動詞 can, could, may, might, must, will の出現回数はカテゴリ毎に特徴がある [11]．例えば news というカテゴリでは，助動詞 will が出現する回数は 389 回で，6 カテゴリ合計の出現回数の約 49 %を占める．しかし，助動詞 can に着目すると, news カテゴリ中では 93 回の出現で全体の 17 %となり，will に比べて少ない．また，hobbies というカテゴリでは，will は 264 回出現し, 全体の 33 %を占めるが，can は,268 回の出現で全体の約 49 %と news カテゴリとは異なる傾向であることが分かる．このように, 助動詞の出現回数にはカテゴリ毎に特徴があると言える．

4.1.1 英文の分類実験

表 1 は文中の助動詞の出現回数を並べた以下の 6 次元ベクトルを使ってその文が所属するカテゴリをある程度の精度で分類することができることを示している．

$$\begin{aligned}(\ & (\text{can の出現回数}), (\text{could の出現回数}), \\ & (\text{may の出現回数}), (\text{might の出現回数}), \\ & (\text{must の出現回数}), (\text{will の出現回数}) \ ) \end{aligned} \quad (1)$$

そこで英文を以下の 3 つの方法で表現し, その英文が所属するカテゴリを分類する実験を行う．

- Sent2Vec を使って文の特徴を 100 次元ベクトルで表現した場合
- Sent2Vec を使って計算した 100 次元のベクトルに，前述の助動詞の出現頻度を表した 6 次元ベクトルを追加し，合計 106 次元ベクトルで文の特徴を表現した場合

[*3] `https://github.com/klb3713/sentence2vec`

表 1　カテゴリ毎の助動詞の出現回数

category	can	could	may	might	must	will
news	93	86	66	38	50	389
religion	82	59	78	12	54	71
hobbies	268	58	131	22	83	264
SF	16	49	4	12	8	16
romance	74	193	11	51	45	43
humor	16	30	8	8	9	13

- 助動詞の出現頻度を表した 6 次元ベクトルを score vector として文と共に ScoreSent2Vec に入力した場合に計算される 100 次元ベクトルで文の特徴を表現した場合

表 2 に Logistic Regression を用いて分類した場合の正解率を示す．それぞれの結果を Sent2Vec, Concatenate, ScoreSent2Vec とした．この表では，PV-DBOW と SPV-DBOW のモデルを使った場合を DBOW，PV-DM と SPV-DM のモデルを使った場合を DM とした．Logistic Regression のプログラムは scikit-learn[12] ライブラリの関数を使用した．正解率は 5 分割交差検定を用いて計算した．

表 2 より, ScoreSent2Vec を使った 100 次元ベクトルによる分類の方が, Concatenate による 106 次元の分類よりも高い正解率を示すことが分かる．これはニューラルネットワークの学習過程で，助動詞だけでなく他の単語を表現するニューラルネットワークにも影響を与えているためと考えられる．具体的には次のように解釈できる．Goldberg ら [13][14] は文脈中に単語が出現する確率から計算される文脈と単語の相互情報量の行列を Matrix Factorizartion したベクトルが，word2vec の計算するベクトル分散表現に相当することを示した．word2vec とほぼ同じ構造を持つニューラルネットワークである paragrph vector についても同様のことが言えると考えられる．ScoreSent2Vec は Matrix Factorizartion される前の行列に，微少変化を加えるシステムと言える．例えば will の出現頻度が強調されるような変化が加えられたとする．その影響は行列の Matrix Factorizartion の結果として will 以外の同じ文脈に出現しやすい他の単語のベクトル分散表現にも影響を与える．例えば tomorrow, future などのベクトル分散表現に影響を与えると考えられる．その結果 will だけでなく，同じ文脈に現れる tomorrow, future などの他の単語の情報も強調されたベクトル分散表現が得られると考えられる．この結果，will の出現回数を単純に数えるよりも，同じ文脈に現れる他の単語の情報も強調されたベクトル分散表現の方が正しく分類する上では有利になっていると考えられる．

この実験のために作成したプログラムのソースコードは `https://github.com/niitsuma/ScoreSent2Vec/blob/master/example-brown-corpus.py` で公開している．

表 2　英文の分類正解率

	DBOW	DM
Sent2Vec	0.426	0.487
Concatenate	0.432	0.484
ScoreSent2Vec	**0.444**	**0.494**

図 7　正しい柴犬の画像の例

4.2　画像分類

次に画像情報が付随する文を分類する実験の結果を示す．具体的には Twitter に投稿された画像を伴う文章を，画像の色ヒストグラムに着目して分類する実験を行う．基本的な実験の構成は藤井らの行った，Twitter に投稿された文章を伴う画像の分類実験 [15] と同様である．

4.2.1　データセット

実験に使用した文章と画像の収集方法について述べる．文章と画像の収集には TwitterAPI 1.1 のキーワード検索を利用した．検出したい対象の名称をクエリとして入力することで，検出対象の名称を含む文章を収集できる．本研究では，検出対象を“ 柴犬 ”とした．またクエリには“ -RT filter:images ” という文字列を追加した．“ -RT ”はリツイートを取得しないためのオプションである．リツイートとは, 他のユーザが投稿したツイートを拡散するツイートのことである．これを取得すると同じ画像を多数収集してしまう可能性があるため, 本実験ではリツイートは取得しない．“ filter:images ”は Twitter の公式アップローダにアップロードされた画像のみを参照するために付与したオプションである．2015 年 7 月 30 日にこの検索を実行し 3,117 件の画像を伴う文章を収集した．この中からランダムサンプリングによって 300 件を抽出し，手動で柴犬の画像が含まれているツイートと含まれていないツイートに分類した．この 300 件のツイートを正解データとして分類を行った．画像の判定基準としては，画像中に柴犬の全身が含まれており，柴犬と認識可能な画像を正しい柴犬の画像とした．図 7 に正しい柴犬の画像の例を示す．なおこの画像は Wikipedia[*4] の柴犬の記事にある画像である．正しい柴犬の画像としないものの例としては, 柴犬のイラストや柴犬以外の物体（他の動物や人）が写っているものがある．300 件の正解データの内, 141 件が正しい柴犬の画像であった．

[*4] `http://ja.Wikipedia.org`

4.2.2 画像の分類実験

4.2.1 で得た 300 件の文章と，その文章に伴う柴犬の画像を 4.1 節と同様に数値ベクトルで表現し，Logistic Regression によって正しい柴犬の画像を含む文章であるかどうかを分類した．5 分割交差検定によってその分類精度を評価する．

文章に付属する画像の情報としては色ヒストグラムを使用する．ただし画素の輝度値が $0, 1, 2, ..., 254, 255$ のそれぞれの値をとる場合を全てヒストグラムにするのではなく，ヒストグラムの bin の数を 256 よりも少なくすることで画像の情報を減らして評価実験を行った．具体的にはヒストグラムの bin の数を $4, 8, 16, 32, 64, 128, 256$ の 7 段階に変化させた場合について評価実験を行った．例えばヒストグラムの bin の数を 4 とした場合は，RGB それぞれの色の輝度を 4 段階に分類してヒストグラムを求める．この場合は RGB それぞれの色の分布が 4 段階 $\times$ 3 色 $= 12$ 次元の特徴が画像の特徴として計算される．本実験では, 分類に用いるベクトルを以下の 4 種類の方法で表し, 画像が正しい柴犬の画像であるか, ないかを分類する．

- Sent2Vec を使って文の特徴を 100 次元ベクトルで表現した場合
- 画像の色ヒストグラムのみを特徴ベクトルとして用いた場合
- Sent2Vec を使って計算した 100 次元のベクトルに色ヒストグラムの値を追加した場合
- 色ヒストグラムを score vector として文と共に ScoreSent2Vec に入力した場合に計算される 100 次元ベクトルで文の特徴を表現した場合

図 8 に上記 4 つの各計算方法について Precision, Recall, F-measure を 5 分割交差検定で算出した結果を示す．それぞれの結果を Sent2Vec, Image Feature Only, Concatenate, ScoreSent2Vec とした．ここで Precision とは, 分類器が “ 正しい柴犬の画像 ” と判断した画像集合の中で手動で分類した正しい柴犬の画像が入っていた割合である．Recall とは, 手動で分類した正しい柴犬の画像集合の中で実際に分類器が “ 正しい柴犬の画像 ” と分類した画像の割合である．F-measure は Precision と Recall の調和平均であり, 次式で求める．

$$\text{F-measure} = \frac{2 * \text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \qquad (2)$$

また, PV-DBOW と SPV-DBOW のモデルを使った場合を DBOW，PV-DM と SPV-DM のモデルを使った場合を DM とした．ScoreSent2Vec と Sent2Vec は全て 100 次元ベクトルで特徴が表現されているが，Image Feature Only と Concatenate は画像特徴の次元が増えると特徴ベクトルの次元も大きくなる点に注意する．例えば bin の数が 256 の場合は，ScoreSent2Vec と Sent2Vec は 100 次元ベクトルであるが，Image Feature Only は $3 \times 256 = 768$ 次元ベクトルであり，Concatenate は $100 + 3 \times 256 = 868$ 次元ベクトルとなり最大の次元数となる．

図 8 の実験結果を見ると，bin の数が 64 以下の場合は (d) のグラフを除くと ScoreSent2Vec が最も良い値となっている．(d) の場合も bin の数が増えて画像の情報を十分に取り込むと他の特徴と同程度の Precision を得られるようになっている．bin の数が 128 以上となった時は，どのグラフでも ScoreSent2Vec は Image Feature Only と Concatenate に劣る結果となっている．これは 100 次元ベクトルである ScoreSent2Vec と比べると倍以上の次元数がある Concatenate は持てる情報量も倍以上あるために生じていると考えられる．Image Feature Only も次元数は ScoreSent2Vec と比べてかなり大きいために，同様の現象が起きている．

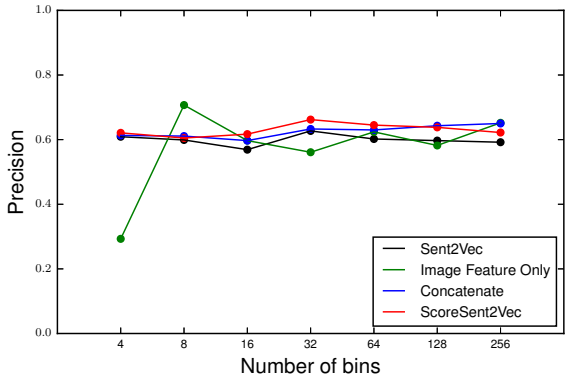
5. 考察

5.1 score vector の条件

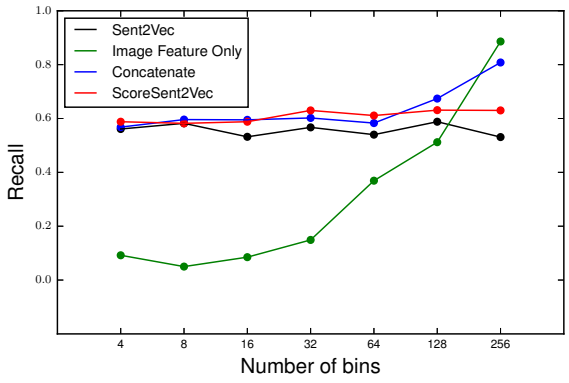
我々は画像の特徴を paragrph vector に埋め込むことを目的として ScoreSent2Vec を開発した．しかし 4.1 節で示したように，数値ベクトルで表現できる情報ならば画像以外の情報でも埋め込むことができる．例えば Amazon の商品レビューならば，そのレビューの評価値と商品の値段をならべた 2 次元ベクトルなども score vector として埋め込むことができる．しかしどんな情報でも埋め込めるわけではなく Paragraph Vector に埋め込める特徴ベクトルすなわち score vector は，次の条件を満たす必要がある．

(1) ScoreSent2Vec の扱える score vector の次元数は実装の関係上，多くて 1000 次元程度までである．例えば学習データが 100 文あるとしたら 100×1000 の浮動小数配列が必要となる実装となっているためである．また 4.2 節の実験結果が示すように，Paragraph Vector よりも大きな次元の特徴ベクトルを Paragraph Vector に埋め込もうとすると，文章の特徴を表す Paragraph Vector の情報が, 埋め込んだ score vector の情報で上書きされて消えてしまう傾向がある．Paragraph Vector の特徴として利用できる妥当な次元数は数百次元程度であり，これを大きく越える情報を埋め込むのは困難となる．この結果，特に画像を扱いたい場合には，画像の全画素を特徴としてそのまま入力するのは困難なシステムであると言える．画像を扱いたい場合には，必ず何らかの特徴抽出プログラムを経由する必要がある．

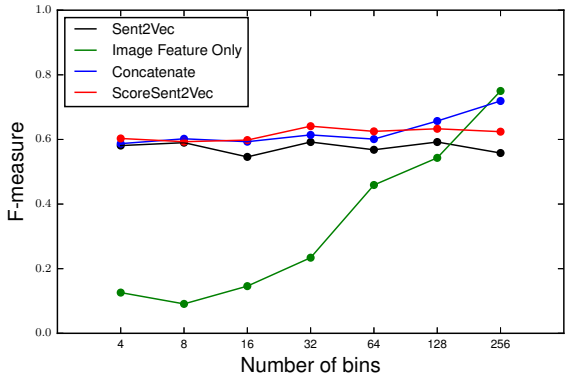
(2) score vector の各ベクトル要素の値はおおよそ-1 から 1 程度の範囲内に収まるように正規化する必要がある．ScoreSent2Vec は score vector を文章にさらに追加さ



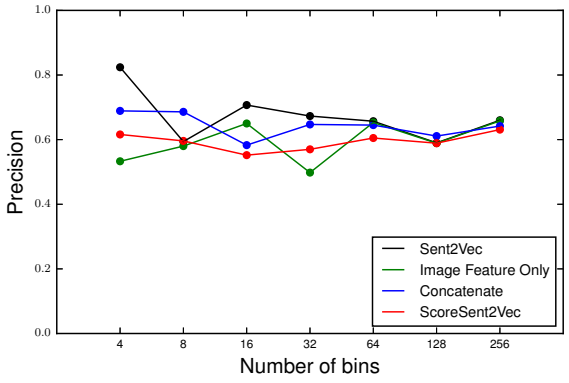
(a) DBOW Precision



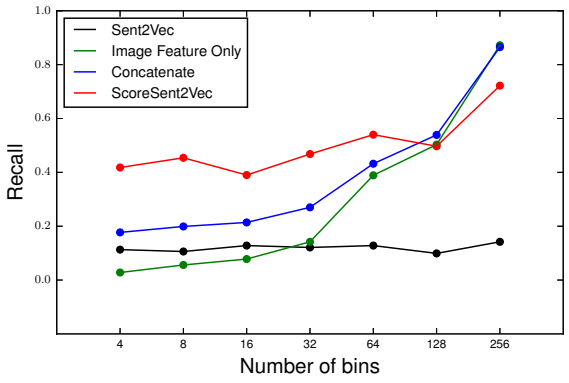
(b) DBOW Recall



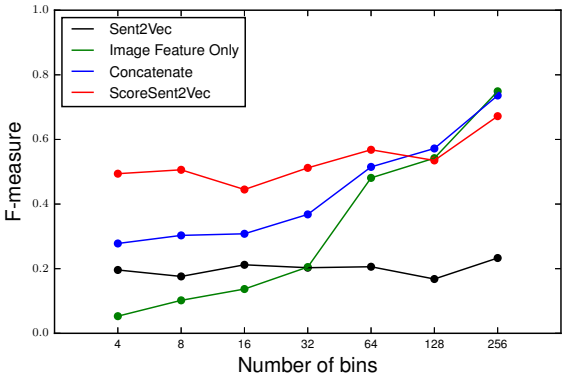
(c) DBOW F-measure



(d) DM Precision



(e) DM Recall



(f) DM F-measure

図 8 追加する画像特徴の次元数と精度の関係

れた単語とみなして取り込むニューラルネットワークである．そのため他の単語のベクトル分散表現と大きく異なる値をとることはできない．安全を考えれば絶対値が 1 以下になる程度に正規化するのが妥当である．

(3) score vector として扱えるのは固定長の数値ベクトルで表現できる特徴だけである．

5.2 DBOW モデルと DM モデルの違い

4.2 節の実験では，PV-DBOW モデルが PV-DM モデルよりも高い精度を実現している．また SPV-DBOW モデルも SPV-DM モデルより高い精度を実現している．どのモデルもニューラルネットワークのバックプロパゲーションによって伝わってきた情報を集めることでベクトル分散表現を計算するモデルである．しかし PV-DBOW モデルと比べて PV-DM モデルはバックプロパゲーションで目的の Paragraph Vector を表すニューラルネットワークに伝わってくる情報が少なくなるような構造となっている．このため PV-DM モデルはあまり高い精度を実現できていないと考えられる．同じことは，ほぼ同じ構造を持つ SPV-DBOW モデルと SPV-DM モデルにも言える．しかし DBOW モデルは，周辺の単語の情報を全て平均してならしてしまうので，必要な情報が消えてしまう可能性もある．例えば特定の手がかり表現が重要となるような場合に，周辺の単語の情報を平均すると，その手がかり表現の情報は消えてしまう可能性がある．これが 4.1 節の実験で DM モデルが有利になった理由と考えられる．

6. まとめ

本研究では，文章と画像の両方の特徴を Paragrach Vector の拡張である ScoreSent2Vec を使って適切に組み合わせる手法を提案した．また ScoreSent2Vec は文章に付随する情報として，英文における助動詞の出現回数と画像情報が利用できることを示した．ScoreSent2Vec はこうした文章に付随する情報を Paragrach Vector のベクトル分散表現の中に埋め込む手法ともみなすことができる．ScoreSent2Vec によって適切な情報が埋め込まれたベクトル分散表現は，Paragrach Vector のみで計算されたベクトル分散表現よりも高い分類精度を達成できる事を実験的に示した．Paragrach Vector と付随する情報の両方を単純に使った場合と比べても，計算される特徴ベクトルの次元があまり変わらない場合には，ScoreSent2Vec によって計算されるベクトル分散表現の方が高い分類精度を達成できる事も実験的に示した．

参考文献

[1] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado and Jeffrey Dean: Distributed Representations of Words and Phrases and their Compositionality, *Advances in Neural Information Processing Systems*, pp. 3111–3119 (2013).

[2] Tomas Mikolov, Kai Chen, Greg Corrado and Jeffrey Dean: Efficient Estimation of Word Representations in Vector Space, *arXiv preprint arXiv:1301.3781* (2013).

[3] Quoc V. Le and Tomas Mikolov: Distributed Representations of Sentences and Documents, *CoRR*, Vol. abs/1405.4053 (2014).

[4] Richard Socher, Alex Perelygin, Jean Y. Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng and C.Potts: Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank, *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Stroudsburg, PA, Association for Computational Linguistics, pp. 1631–1642 (2013).

[5] 岩井美樹, 二宮崇：word2vec に基づく述語項構造の分布表現獲得，言語処理学会　第 21 回年次大会　発表論文集, pp. 75–78 (2015).

[6] 小西友七：英語基本動詞活用辞典，研究社 (1980).

[7] 黒崎優太，高木友博：Word2Vec を用いた顔文字の感情分類，言語処理学会　第 21 回年次大会　発表論文集，pp. 441–444 (2015).

[8] 中野滋徳，足立顕，牧野武則：提題表現に基づく重要段落抽出，情報処理学会研究報告．NL，自然言語処理研究会報告，Vol. 162, pp. 159–166 (2004).

[9] 佐藤元紀，伊藤孝行：Paragraph Vector と多層パーセプトロンを用いた有害文書の分類手法，情報処理学会第 77 回全国大会，pp. 165–166 (2015).

[10] W.N.Francis and H.Kucera: *Brown Corpus Manual*, Brown University (1964/1979).

[11] Steven Bird, Ewan Klein and Edward Loper: *Natural Language Processing with Python*, O'Reilly Media (2009).

[12] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E.: Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research*, Vol. 12, pp. 2825–2830 (2011).

[13] Goldberg, Y. and Levy, O.: word2vec Explained: deriving Mikolov et al.'s negative-sampling word-embedding method, *arXiv preprint arXiv:1402.3722* (2014).

[14] Levy, O. and Goldberg, Y.: Neural Word Embedding as Implicit Matrix Factorization, *Advances in Neural Information Processing Systems 27* (Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N. and Weinberger, K., eds.), Curran Associates, Inc., pp. 2177–2185 (online), available from ⟨http://papers.nips.cc/paper/5477-neural-word-embedding-as-implicit-matrix-factorization.pdf⟩ (2014).

[15] 藤井一哉, 新妻弘崇, 太田学：Web 上の画像の周辺テキストを用いた自動画像アノテーション，*DEIM Forum 2015 F6-4*, pp. 1–7 (2015).