The 32nd Annual Conference of the Japanese Society for Artificial Intelligence, 2018

文字 Embedding を使用したニューラルネットワークによる顔文字の原形推定

A Method to Estimate the Original Form of Kaomoji Using Neural Network with Character Embedding Function

奥村 嶺　　奥村 紀之
Rei Okumura　Noriyuki Okumura

明石工業高等専門学校
National Institute of Technology, Akashi College

This research focuses on Kaoomji (Asian style emoticons) used in text-based communication. We analyze Kaomoji that are made from particular characters' sequence (original shapes). In this paper, we construct a neural network with embedding function to estimate the original form of Kaomoji. As a result, we find out that our proposed method can estimate the original form of Kaomoji with 0.746 of accuracy ratio.

1. はじめに

光通信や 4G/LTE 通信などの高速インターネット普及に伴い，オンラインでのコミュニケーションは文字情報だけでなく，画像や音声，動画などのメディアも使用可能なサービスが増えてきている．しかし，依然として SNS 等における文字情報の必要性は高く，LINE[*1] でのメッセージのやり取りや Facebook[*2] のコメント，Twitter[*3] のリプライなど，文字列に依存したコミュニケーションは活発に交わされている．

ところが，日本語でのコミュニケーションでは，空気を読む，雰囲気に合わせるといった，察する能力が求められる．このようなハイコンテクスト（抽象的な表現を好む）言語であるとされる [Hall 76] 日本語は，英語やドイツ語のような多くの内容を明記して情報を伝達する傾向のある言語と比べ，文字列のみのコミュニケーションと両立させることが困難である．一例として，不用意な発言 1 つで炎上 [*4] などのトラブルに発展してしまうことがある．これは，書き手の言葉以外の情報（表情等）が，正確に受け手に伝わらないことで起こる問題である．例えば受け手が，書き手の意思とは反する字面通りの解釈をしてしまったり，書き手の言葉の奥にあるより伝えたい情報を読み違えたりして引き起こされる．

このように，日本語でのコミュニケーション，特に対面での会話では，話し手は多くを語らず，身振りや表情を交えることで聞き手に言葉以外の情報を伝える．そのため，文字列でのコミュニケーションでは，受け手は文字情報のみから全ての情報を読取る必要があり，読み手の負担が増大してしまう．読み手の負担を軽減するには，書き手が過不足無く情報を記載することが求められる．しかし，過不足なく理解しやすい文章を書くことは決して容易ではなく，そのようなことに慣れていない日本語話者にとっては極めて深刻な問題である．

書き手と読み手の負担を軽減させる一例として，顔文字が挙げられる．顔文字は＼(^o^)／のように，表情や動作を文字列（記号列）によって表したものである．この顔文字例からは，嬉しい表情 ((^o^)) と両腕を上に挙げる動作 (＼／) の情報を得ることできる．このように，顔文字を用いることで，文章では直接伝えられない表情や動作を視覚的に伝達することが可能となる．

しかし，奥村によれば現在，顔文字の種類は 10 万種を超えており，今後も増加すると予想される [奥村 16c]．そのため，顔文字のリストを大量に保持するだけでは不十分であり，未知の顔文字に対するアプローチを検討する必要がある．さらに，顔文字は言語として成立している文字列の解析とは異なり，ありとあらゆる記号列が構成要素となる．このことから，すべての顔文字に品詞等のルールを制定することが困難である．そこで新たなアプローチとして，顔文字をある文字列（原形）から派生したものであるとみなし，原形に付随するパーツの傾向から確率的に顔文字の持つ情報を推定することが重要であると考えられる．

我々は，前述の新たなアプローチを研究する前段階として，とある顔文字（以下「元文字列」と呼称）から，その顔文字の派生元である文字列（以下「原形」と呼称）を推定する検証を行っている．我々は，この推定を顔文字の原形推定と呼称している．推定には，Cosine 類似度による手法や，サポートベクトルマシン（SVM）による手法など，多種多様なアプローチがある．本実験ではニューラルネットワークを用いた手法を扱う．従来手法 [奥村 17] では，顔文字を数値化する手法に問題があり，優良な結果が得られなかった．そこで本稿では，顔文字をニューラルネットワークで用いるにあたって Embedding を導入し，検証した結果について述べる．

2. 関連研究

本研究に関連して，我々は顔文字へのアノテーションを進めている [奥村 16c]．顔文字を包括的に扱い，確率的なモデルとして活用する為には，大規模な辞書が必要となる．顔文字は様々な記号から作られる可能性があるため，記号に着目して多元的に分析する必要がある．

顔文字の抽出に関する研究として，Tanaka らの研究では，機械学習によって顔文字を抽出を試みている [Tanaka 05]．また，文中からの顔文字抽出に関する研究として，Bedrick らの手法がある．Bedrick らは，隠れマルコフモデル (HMM) によって記号列を抽出し，確率的文脈自由文法 (PCFG) に基づく評価法によって顔文字の候補を選別する手法を提案している [Bedrick 12]．

また，テキストに含まれる顔文字の抽出に関する研究としては，Ptaszynski らの研究がある [Ptaszynski 10]．Ptaszynski らの CAO システムでは，顔文字における目-口-目の並び

連絡先: 奥村 嶺，明石工業高等専門学校，e1371@s.akashi.ac.jp

∗1 https://line.me
∗2 https://www.facebook.com
∗3 https://twitter.com
∗4 非難や誹謗中傷のコメントが殺到して収拾がつかなくなる状態

(Triplet) に着目した顔文字抽出を試みている．このシステムでは，1 万種の顔文字にデータベース規模で対応し，90% 以上の精度で抽出可能だとしている．さらに自動拡張によって 300 万種に対応可能であると述べられており，これは顔文字の研究の中では非常に規模の大きいものである [Ptaszynski 12]．奥村らは，Ptaszynski らの Triplet に加え，輪郭を表現する文字列を加えたものを顔文字の原形として定義 [奥村 16b] しており，本実験の原形はこれに準ずる．さらに，同じ顔文字の原形を推定する研究に関しては，手法としてサポートベクトルマシン (SVM) を用いた研究 [奥村 16a] が参考になる．

3. 顔文字の原形推定

先行研究 [奥村 16c] より，顔文字の原形はおよそ 3,000 種類発見されている．また，顔文字と認識された任意の文字列とその原形のペアが 43,733 組分類されている．

本稿では，顔文字と認識された任意の文字列（以下「元文字列」と略す）から，その顔文字の原形をニューラルネットワーク (Neural Network，以下「NN」と略す）によって推定する検証の進行状況を述べる．図 1 に顔文字の原形推定の例を示す．

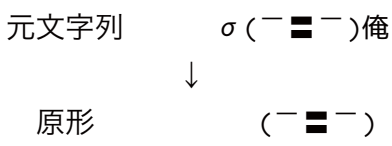

図 1　顔文字の原形推定の例

実験環境としては，Python(3.5.2)，Chainer(2.0.0)，CUDA(8.0.44) を用い，評価指標としては，式 (1) 示す分類正解率 (Accuracy) を用いた．分類正解率は 0 から 1 の値を取り，値が 1 に近いほど優良とする．

$$\text{分類正解率} = \frac{\text{正解した事例数}}{\text{検証事例数}} \tag{1}$$

4. 従来の文字列の数値化と問題点

NN では数学的な計算によって学習を進めていくため，文字列を文字情報として直接用いることができない．そのため，文字列を何らかの数値に変換する必要がある．これまでの検証 [奥村 17] では，顔文字を次の通りに数値化している．

原形の数値化

原形については，原形の種類ごとに 0 から 3,109 のラベルを付与する．具体的には，全ての原形に対して，原形の種類ごとに固有の数値を順に割り当てる．

元文字列の数値化

元文字列は，元文字列に使用されている文字だけでなく，文字の並びも考慮して数値化する必要がある．事前調査により，Bigram，Trigram，SkipBigram（一文字飛ばしの Bigram）において，高い分類正解率が得られたため，この 3 種類の N-gram に着目している．具体的な手順を次に示す．

1. 全ての元文字列に対して Bigram 要素，Trigram 要素，SkipBigram 要素を抽出
2. 各 N-gram 要素を数値に変換 (後述※ 1)
3. 各 N-gram ごとに得られた数値列を一列になるように結合

また，上記の※ 1 の手順を Bigram を例として以下に示す．

1. Bigram 要素全てに対して，その要素が既にデータベースにあるか調べる．
2. 要素がデータベースにない場合，新たな重複しない値を要素に割り当てる．また，データベースに登録する．
3. 要素がデータベースにある場合，その数値を参照し，割り当てる．

以上の工程を経て，従来の数値化では表 1 に示すようなデータを作成していた．

このような数値化手法では，各抽出要素の数値の差が各抽出要素の意味の異なり度合いと一致するように学習が行われる．そのため，数値の大小と N-gram の特徴を対応付けることができず，NN の学習の弊害となっていることが推察される．

5. 提案手法

前節に示した，数値化の問題の解決策として，我々は Embedding を用いた検証を行った．Embedding とは，単語や文字，概念などを分散表現と呼ばれるベクトル空間に埋め込むことである．単語を分散表現で表す手法 (Word Embedding) の代表的な例として，Mikolov らの Word2Vec[Mikolov 13] が挙げられる．

我々が NN に組み込んだ Embedding 層は，文字を識別するユニークな数値 (スカラ) を入力とし，固定長のベクトルを出力とする．本実験では出力ベクトルの大きさ (長さ) を 100 としているため，この Embedding 層は各文字を 100 個の要素を持ったベクトルに変換する．この変換は NN の学習によって動的に変化する．

表 1　データ加工の一例

元文字列	σ(￣〓￣)俺？							
Bigram	抽出要素	σ(	(￣	￣〓	〓￣	￣)	)俺	俺？
	数値化	9900	4748	10805	9682	1672	19532	4896
Trigram	抽出要素	σ(￣	(￣〓	￣〓￣	〓￣)	￣)俺	)俺？	
	数値化	17567	18223	31134	42804	18325	61194	
SkipBigram	抽出要素	σ￣	(〓	￣￣	〓)	￣俺	)？	
	数値化	14250	8836	34142	26975	13545	17352	
原形	(￣〓￣)			ラベル	266			

図 2 に実験で用いた NN の構造を示す．

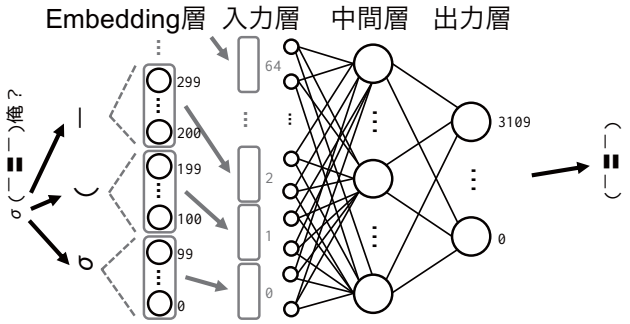

図 2　Embedding を用いる NN

ここで，図 2 に明記している各数値の根拠を示す．

- Embedding 層から出力されるベクトルが 65 個である理由は，実験に使用したデータの元文字列の長さの最大が 65 だからである．元文字列が 65 よりも短い場合は，文字なしに相当する分散表現を出力する．
- 出力層が 3,110 ユニットである理由は，実験に使用したデータの原形の種類数が 3,110 だからである．

5.1　使用したデータ

本項では，使用した学習データと検証データについて示す．元データにある 43,733 組の元文字列と原形のデータには，表 2 に示すように，原形の種類に応じて元文字列の個数にばらつきがあり，100 種類以上の元文字列がある原形もあれば，1 つの元文字列しかない原形もある．このばらつきは，NN の学習データを作成する際に，選択される原形の種類に偏りが生じるような弊害となる．

我々はこの問題を解決するために，3,110 種類ある原形全てが 100 種類の元文字列を持つようなデータ（LargeKaomoji-Data:LKD）を作成し，これを学習データとして用いた．作成にあたっては，全原形それぞれに対し復元抽出を行った．

評価データは，元データから元文字列と原形の組を 1,000 回復元抽出したものを 10 セット用意し使用した．

5.2　従来手法との比較

従来手法で使用されている NN と比較するため，中間層のユニット数や活性化関数，最適化手法を以下の通りに設定し，学習を行った．

- 中間層のユニット数は 10,000
- 活性化関数は式 (2) に示す Sigmoid 関数
- 最適化手法は Adam

$$Sigmoid(x) = \frac{1}{1+\exp(-x)} \tag{2}$$

図 3 に学習の経過 (Epoch) と分類正解率 (Acc) の変化を示す．また，図中の破線は学習データに対する分類正解率の推移で，実線は評価データに対する分類正解率の推移である．図 3 より，評価データに対する推移に着目すると，20Epoch の学習で既に分類正解率 0.7 を超え，70Epoch 学習したところで最大の分類正解率である 0.746 に達している．また，従来手法では 280Epoch の学習で分類正解率が 0.51 であることから，提案手法は従来手法と比べ少ない学習回数でより優良な分類正解率を示すことが見て取れる．

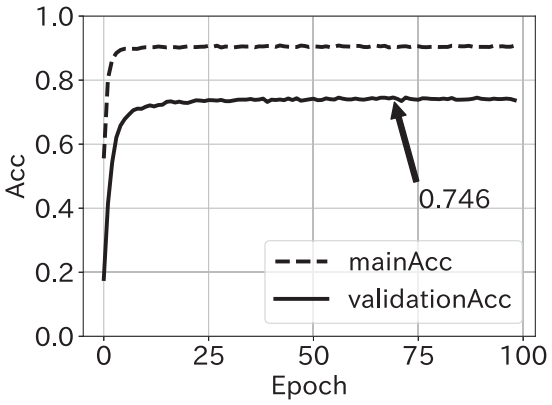

図 3　Embedding 層を追加した NN での学習回数と分類正解率の変化

表 2　元データに含まれる原形と元文字列の個数

原形	元文字列の事例数	事例 (元文字列)
(・_・)	1,032	ドキ（*・ __ ・*）ドキ C= C= ＼(;・_・)/ ーー＝＝（/_・)/＝＝≡≡卍スカッ
(-_-)	610	”/(;-_-) ｲﾃﾃ･･･ ヾ(-_-)ｼ…ｱ ﾖｲﾖｲ！ ＼(-__________-;)／
(・ω・)	589	(‘・ω・´)ノ ヾ(・ω・`)ノ ←・・・・・・(・ω・.) ジーッ
U=ｴ=U	1	ブルブル（(U= ｴ =U)）トイレ行きたい
(ε(ｴ)ε)	1	○-○へ キョロC(ε(ｴ)ε)つ)))(((C(3(ｴ)3) つキョロ
(|￣|_|￣|)	1	拉致 |Φ|(|￣|_|￣|)|Φ| 監禁！

5.3 従来手法との差異と考察

従来手法では，元文字列を Bigram, Trigram, SkipBigram の各要素に分解し，それぞれに固有の数値を与えることで，NN に入力し学習していた．そのため従来手法では，各 N-gram 要素と入力層にインプットする数値は固定されており，学習において変化しない．しかし，提案手法では，各文字を分散表現 (固定長ベクトル) に変換する過程も NN に組み込む．その結果，各文字のベクトル表現も学習によって改善させることができ，分類正解率が向上したと考えられる．

5.4 学習データに対する分類正解率

本来，学習データは学習に用いるデータであるため，学習データに対する Accuracy は限りなく 1 に近づくことが学習の理想である．しかし，図 3 の破線に着目すると，学習の進行に伴って分類正解率が 0.9 から変動していないことが見て取れる．これは，顔文字の原形抽出ルールが関係している．

我々が準じている顔文字の原形抽出ルールでは，元文字列が複数の原形を含んでいる場合，その全ての原形を抽出するルールとなっている．しかし，提案手法の NN では，1 つの元文字列に対して 1 つの原形のみ推定する構造となっているため，複数の原形を持つ事例では，一定の確率で誤りだとカウントされてしまう．複数の原形を持つ事例は，学習データには約 30% ほど含まれており，これが学習データに対する分類正解率が 0.9 から向上しない原因だと考えられる．表 3 に複数の原形を持つ事例と，1 つの原形しか持たない事例を示す．

6. まとめ

本稿では，顔文字の原形推定に関して，Embedding を用いたニューラルネットワークでのアプローチの有用性について述べた．従来手法 [奥村 17] では，ベースラインとしての Cosine 類似度による推定が苦手とする，**llllll(-ë-;)llllll ずーん** のような原形に関係の無い付随パーツが大量に存在する顔文字（元文字列）に対して優秀であるとしながらも，顔文字の数値化に問題があり，分類正解率は 0.51 に留まっていた．

我々の提案手法では，表 4 に示すように，Embedding 層を追加したニューラルネットワークを用いることにより，従来手法よりも少ない学習回数で，優良な分類正解率 0.746 を得た．

表 4　従来手法と提案手法の学習回数と分類正解率

	学習回数 (Epoch)	分類正解率
従来手法	280	0.51
提案手法	70	0.746

今後の課題として，複数の原形を持つ元文字列への対応が挙げられる．現在の FFNN では，1 つの元文字列に対して複数の原形を推定することは不可能であるため，改良が必要である．一例として，新たに元文字列から原形の個数を推定するモデルを構築し，FFNN が原形を推定する際のヒントにする手法が考えられる．また，提案手法の FFNN を用いて原形から原形を推定した結果の Accuracy は 0.066 であり，今後の検討が必要である．

謝辞

本研究は JSPS 科研費 15K21592，高専連携教育研究プロジェクト 課題番号 2309 の助成を受けたものである．

参考文献

[Bedrick 12] Bedrick, S., Beckley, R., Roark, B., and Sproat, R.: Robust kaomoji detection in twitter, in *Proceedings of the Second Workshop on Language in Social Media*, pp. 56–64, Montreal, Canada (2012), Association for Computational Linguistics

[Hall 76] Hall, E. T.: *Beyond Culture*, Anchor (1976)

[Mikolov 13] Mikolov, T., Chen, K., Corrado, G., and Dean, J.: Efficient Estimation of Word Representations in Vector Space, *CoRR*, Vol. abs/1301.3781, (2013)

[Ptaszynski 10] Ptaszynski, M., Dybala, P., Rzepka, R., and Araki, K.: CAO: A Fully Automatic Emoticon Analysis System Based on Theory of Kinesics, *Affective Computing, IEEE Transactions*, Vol. 1, No. 1, pp. 46–59 (2010)

[Ptaszynski 12] Ptaszynski, M., Maciejewski, J., Dybala, P., Rzepka, R., Araki, K., and Momouchi, Y.: Speech, Image, and Language Processing for Human Computer Interaction - Science of Emoticons: Research Framework and State of the Art in Analysis of kaomoji-type Emoticons., *IGI Global* (2012)

[Tanaka 05] Tanaka, Y., Takamura, H., and Okumura, M.: Extraction and classification of facemarks, in *Proceedings of the 10th International Conference on Intelligent User Interfaces*, IUI'05, pp. 28–34, New York, NY, USA (2005), Association for Computing Machinery

[奥村 16a] 奥村紀之 F 分類器による顔文字の原形推定, 信学技報, Vol. 116, No. 379, pp. 93–96 (2016), NLC2016-37

[奥村 16b] 奥村紀之 F 感情抽出のための顔文字の原形推定, 信学技報, Vol. 116, No. 78, pp. 1–4 (2016), NLC2016-1

[奥村 16c] 奥村紀之 F 言語解析のための大規模顔文字辞書, 人工知能学会第 30 回全国大会 (2016)

[奥村 17] 奥村嶺 F ニューラルネットワークによる顔文字の原形推定, 2017 年度 人工知能学会全国大会（第 31 回）(2017)

表 3　推定の難易度から見た事例の一例

	元文字列	原形
1 つの原形しか持たない事例	┌|T_T|┘♪ └|T_T|┐♪ ┌|T_T|┘ **テケテケテッテ** (((-_-(-_-;=(-_-;=;-_-)=;-_-)-_-))) **分身** **ア**（・ ．・）**リ**（・ ．・）**ガ**（・ ．・）**ト**（・ ．・）**ウ**（・ ．・）	|T_T| (-_-) (・.・)
複数の原形を持つ事例	**モジ**(((*Φ ｗ Φ *)(* Φ ３Φ*))) **モジ** **ﾗﾌﾞﾗﾌﾞ**(((*--)￣(**エ**)￣) ♪ **ｳﾃﾞﾏｸﾗ**♪ (*￣-(-*) ヽｽﾔｽﾔ	(**Φ** ３ **Φ**), (**Φ** ｗ **Φ**) (--), (￣ (**エ**)￣) (--), (￣ -￣)