IEICE TRANS. INF. & SYST., VOL.E101–D, NO.4 APRIL 2018

PAPER Special Section on Data Engineering and Information Management

Semantically Readable Distributed Representation Learning and Its Expandability Using a Word Semantic Vector Dictionary**

Ikuo KESHI[†*a)], **Yu SUZUKI**[†], ***Members***, **Koichiro YOSHINO**[†], ***Nonmember***, ***and*** **Satoshi NAKAMURA**[†], ***Member***

SUMMARY The problem with distributed representations generated by neural networks is that the meaning of the features is difficult to understand. We propose a new method that gives a specific meaning to each node of a hidden layer by introducing a manually created word semantic vector dictionary into the initial weights and by using paragraph vector models. We conducted experiments to test the hypotheses using a single domain benchmark for Japanese Twitter sentiment analysis and then evaluated the expandability of the method using a diverse and large-scale benchmark. Moreover, we tested the domain-independence of the method using a Wikipedia corpus. Our experimental results demonstrated that the learned vector is better than the performance of the existing paragraph vector in the evaluation of the Twitter sentiment analysis task using the single domain benchmark. Also, we determined the readability of document embeddings, which means distributed representations of documents, in a user test. The definition of readability in this paper is that people can understand the meaning of large weighted features of distributed representations. A total of 52.4% of the top five weighted hidden nodes were related to tweets where one of the paragraph vector models learned the document embeddings. For the expandability evaluation of the method, we improved the dictionary based on the results of the hypothesis test and examined the relationship of the readability of learned word vectors and the task accuracy of Twitter sentiment analysis using the diverse and large-scale benchmark. We also conducted a word similarity task using the Wikipedia corpus to test the domain-independence of the method. We found the expandability results of the method are better than or comparable to the performance of the paragraph vector. Also, the objective and subjective evaluation support each hidden node maintaining a specific meaning. Thus, the proposed method succeeded in improving readability.

key words: *distributed representation, word semantic vector dictionary, paragraph vector, word2vec, Twitter, sentiment analysis*

1. Introduction

Distributed representations named word2vec and paragraph vectors, computed using simple neural networks with context information as features, have been widely used [2]–[5]. The paragraph vectors achieved state-of-the-art results on sentiment analysis at the time of publication [5]. The problem in engineering with the distributed representations of words and paragraphs is that the meaning of the distributed representations is difficult to understand. Thus, quality assessment of distributed representation learning has no choice but to rely on task accuracies using learned distributed representations.

In contrast to the distributed representations obtained by learning, the authors proposed word semantic vectors, constructed using a human expert with context information as features [6]. The word semantic vector expresses the relationship between a word and feature words as a binary value that is related or unrelated. The feature word corresponds to each dimension of the word semantic vector, and it consists of 266 conceptual classifications. The core words are composed of 20,330 important words extracted using frequency analysis of Japanese newspapers and encyclopedias. The word semantic vector dictionary is a dictionary listing feature words related to the core words.

We proposed an integration method to learn feature words expanded using the word semantic vector dictionary with a paragraph vector model to solve the problem of word sparsity in Twitter [7]. The problem of word sparsity means that many of the words in short text to be analyzed are not included in the vocabulary learning distributed representations. The integration of the word semantic vector dictionary and paragraph vector learning showed that the accuracy of sentiment analysis improves by learning context information of a particular domain with expanded feature words using the dictionary. We showed that expanded feature words for tweets can be used for error analysis of sentiment analysis, but it was still difficult to read the features of distributed representations.

This paper proposes a new method of automatically learning readable distributed representations using the paragraph vector models based on the word semantic vector dictionary [1]. Word semantic vector dictionaries are more like distributed representations rather than semantic lexicons like WordNet [8] and FrameNet [9] because each core word is defined as a fixed-length dense vector. In this paper, the 266 feature words are taken as the hidden nodes of each model. Then, the initial weights between the input word and each hidden node, which is a seed vector, are given based on the dictionary. The difference between the proposed method which is called our method and the conventional method which is the paragraph vector [5] is whether the seed vector is generated from the dictionary reflecting human knowledge or is random values. Our method uses the paragraph vector models [5] for unsupervised learning, except for the difference in the seed vector.

The main purpose to realize the readability of

Manuscript received June 27, 2017.
Manuscript revised November 6, 2017.
Manuscript publicized January 18, 2018.

[†]The authors are with Nara Institute of Science and Technology, Ikoma-shi, 630–0192 Japan.
[*]Presently, the author is with Fukui University of Technology, Fukui-shi, 910–8505 Japan.

[**]A short version of this paper was presented at WI2017 [1].

a) E-mail: keshi@fukui-ut.ac.jp

DOI: 10.1587/transinf.2017DAP0019

distributed representations is to enable social media mining with specific concepts by unsupervised learning if it can make clear which hidden nodes (specific concepts) are strongly bound to words and documents. For example, we can reduce the number of not-related tweets using the specific concepts automatically if each dimension of distributed representations is readable.

In order to verify the hypothesis that the meaning of each hidden node is maintained by giving an appropriate seed vector based on the dictionary, the following tests are conducted. First, we verify the hypothesis using a single domain benchmark for Japanese Twitter sentiment analysis in Appendix A. Second, we evaluate the expandability of our method using a diverse and large-scale benchmark in Appendix B. Moreover, we test the domain-independence of our method using the Wikipedia corpus.

Using the single domain benchmark, we show the readability of word and document embeddings in three ways: First, we evaluate the readability of word embeddings using the correlation coefficient between the seed vector and the learned word vector. Even in new words, the meaning of the hidden node is maintained to some extent. Second, the task accuracy of our method is better than the performance of the conventional method. Third, we present an evaluation of the readability of document embeddings in a user test. A total of 52.4% of the given feature words were related to tweets where one of the paragraph vector models learned the document embeddings for the top five weighted hidden nodes.

We demonstrated the readability of the document embeddings in a single domain benchmark. However, testing the expandability of the proposed method is important. We improved the dictionary for this purpose. The dictionary consists of 264 feature words–not 266–and 20,330 core words. The diverse and large-scale benchmark consists of 38,576 tweets labeled positive or negative for eight categories. The evaluation results show that the performance of sentiment analysis is better than or comparable to the conventional method's while maintaining the correlation between the word vectors learned with 3 million tweets and the seed vectors based on the dictionary.

Moreover, to evaluate the domain-independence of our method using the Wikipedia corpus and a Japanese word similarity dataset, we found that synonyms have similar vectors while the top five weighted feature words of the core word after learning are related to the core word. Therefore, our method improves the readability of the distributed representations, which are the weights of each hidden node for words and paragraphs. The contributions of this paper are as follows.

- We assigned each feature word in the dictionary to one hidden node in neural networks and initialized the weights of the neural networks by recursive extension of the dictionary. When training unlabeled text data using neural networks, each learned feature is usually a black box. Because the proposed method learns the weights of preselected feature words, we demonstrated it improves the readability of word and document embeddings.
- We evaluated the expandability of the proposed method by conducting a sentiment analysis task using a diverse and large-scale benchmark and a word similarity task using the Wikipedia corpus.

2. Related Work

Research on the integration of external semantic lexicons and distributed representation learning has been active. The following three research directions are being studied.

- **Pre-processing.** Feature word expansion on Twitter of our previous proposal is part of this direction [7]. Tweet2vec [10] trained the CNN-LSTM encoder-decoder model on 3 million randomly selected tweets populated using data augmentation techniques, which are useful for controlling generalization error for a deep learning model. Data augmentation techniques refer to replicating tweets and replacing some words with their synonyms using WordNet [8].
- **Learning process.** RC-NET [11] is built upon the Skip-gram model [3], the objective function of which is extended by incorporating both the relational knowledge (like is-a, etc) and the categorical knowledge (like synonyms) as regularization functions. Bollegala et al. proposed a global word co-occurrence prediction method [12] using the semantic relations in WordNet as a regularizer [13]. Our proposal in this paper is part of this direction.
- **Post-processing.** Retrofitting [14] is a technique for fitting learned word vectors to semantic lexicons. In this paper, we used this technique to create initial weights of core words.

Experiments have shown that the precision of distributed representations of words has qualitatively improved the best in [13] and that the accuracy of sentiment analysis has improved in [10]. Both showed that they were state-of-the-arts techniques using standard datasets on word similarity and sentiment analysis. Similar studies have been done using topic models based on LDA [15]. Topical word embeddings (TWE) [16], in which "topical word" refers to a word taking a specific topic, have been proposed to measure contextual word similarity by extending the Skip-gram model [3]. Also TWE outperformed the Skip-gram model in word similarity tasks. TWE is also applied to tweet topic classification tasks and performs better than paragraph vectors [17]. However, no reports on the relevant literature describe an attempt to give meaning to each hidden node.

One model of paragraph vectors (PV-DBOW) [5] uses pre-trained word embeddings that reportedly improve task performance [18]. Although this paper shows the possibility of learning proper document embeddings with good initialization of word embeddings, it does not demonstrate the possibility of interpretation of hidden nodes.

The point to emphasize is that topic models, for example, can be used to assign topic numbers and related keywords to each tweet, but people cannot understand the meaning of large weighted features of tweet embeddings using any of the conventional methods.

3. Proposed Method

This paper presents a test of the hypothesis that the meaning of each hidden node is maintained even after (the unsupervised learning part of) three approaches: ①assigning specific meaning to each hidden node, ②giving the strength of the semantic and associative relationship with each hidden node as the initial weights of important words, and ③unsupervised learning using word2vec and paragraph vector models. The neural network learns the concept automatically for the hidden layer. Thus, the weights may adapt to the context with the concept maintained by pre-setting the appropriate conceptual classification to be learned to the nodes of the hidden layer and by giving the suitable initial value.

3.1 Word2vec and Paragraph Vector Models

Figure 1 presents two variants of word2vec and paragraph vector models [5]. The distributed memory model of paragraph vectors (PV-DM) predicts the target word vector of the next word w (t) from the context vector obtained by adding a paragraph ID to input words within the context window. The continuous bag of words (CBOW) of word2vec does not add the paragraph ID to the input layer, but it is fundamentally the same as the PV-DM.

The paragraph vector with a distributed bag of words (PV-DBOW) learns the paragraph vector to predict the context word vectors of randomly selected surrounding words within the context window. Skip-gram of word2vec is used to learn the vectors of the target words in the PV-DBOW. In Skip-gram, the target word vector is learned so that the inner product of the target word vector and the context word vector of the surrounding words is larger than the inner product of the context word vector of words other than the surrounding words.

3.2 Word Semantic Vector Dictionary

We selected 266 conceptual classifications that belong to six major classes and 29 upper concepts as feature words in the word semantic vector dictionary [6], as presented in Table 1. For core words, we selected 20,330 words from encyclopedias, newspapers, websites, instruction manuals, and Kansei words.

A human expert assigned feature words to each core word based on the following criteria. Feature words were assigned from a logical and associative relationship. The logical relationship refers to those in which the feature words have direct relevance for core words, as shown in Table 2. The associative relationship refers to those in which feature words are related to core words by association, as shown in Table 3.

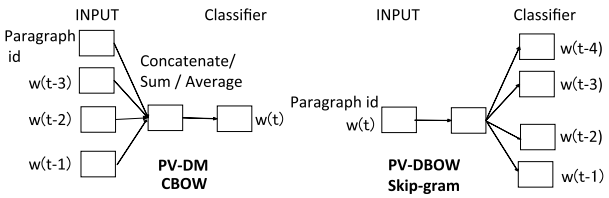

Fig. 1 Word2vec and paragraph vector models.

Table 1 Classification of feature words.

Six large classifications	Examples of 29 upper concepts	Examples of 266 feature words
Human·Life	Human	Human, Name, Male, Female, Child
	Creature	Animal, Bird, Insect, Microbe, Plant
Human environment	Artificiality	Tool, Mechanical·Component, Building
	Traffic·Communication	Communication, Traffic·Transportation
Natural environment	Area	Place name, Country name, Japan, City
	Nature	Land, Mountain, Sky, Ocean
Abstract concept	Spirit·Psychology	Sense, Emotion, Happiness, Sadness
	Abstract concept	State·Aspect, Change, Relationship
Physics·Substance	Motion	Motion, Halt, Dynamic, Static
	Physical characteristics	Warmth, Weight, Lightness, Flexible
Civilization·Information	Humanities	Race, Knowledge, Speech
	Science	Mathematics, Physics, Astronomy

Table 2 Grant criteria by logical relationship.

Logical relationship	Core words	Feature words
Class inclusion	Autumn	Season
Synonym relationship	Idea	Thought
Part–whole relationship	Leg	Human body

Table 3 Grant criteria by associative relationship.

Core words	Feature words
Love	Kindness, Warmth
Up	Economy, Video
Leg	Car, Traffic·Transportation

3.3 Model Setting for Testing the Hypothesis

In this section, we describe the setting to encode the initial weights of the core words based on the strength of the relationship with each feature word, and we describe our test of the hypothesis using Skip-gram as an example.

A method has been proposed for generating a word vector by recursively expanding a definition sentence for a word in a dictionary [19]. The word semantic vector dictionary can be regarded as defining a core word with 266 types of feature words. Because feature words are also core words, recursive extension is necessary. However, convergence occurs when the feature word is expanded several times because the definition sentence of the core word is limited to 266 words. Also, a method has been proposed for retrofitting word vectors according to related words in a dictionary [14]. In that method, we generate a seed vector of the core word by recursively expanding the dictionary using retrofitting tools[†].

When building a vocabulary from the corpus, the initial vectors of the following two kinds are created first.

[†]https://github.com/mfaruqui/retrofitting

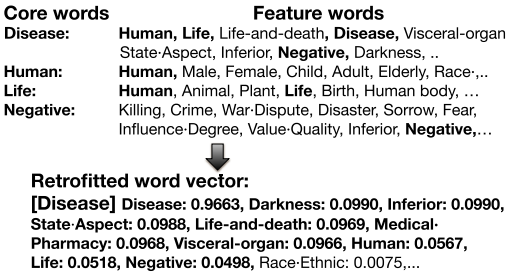

Fig. 2 Example of retrofitting "Disease."

- The 266 feature words are added to the vocabulary as one-hot vectors with dimensions corresponding to respective feature words set to 1.
- Other initial word vectors including core words extracted from the corpus are 266-dimensional zero vectors.

The retrofitting algorithm aims at bringing word vectors closer to the relationship of the word entries of the lexicon as post-processing of learning of word vectors [14]. We applied this algorithm for retrofitting the aforementioned initial word vectors, which are 266-dimensional one-hot or zero vectors, into the word semantic vector dictionary. The retrofitting algorithm is shown as the following online update [14]:

$$\mathbf{q_i} = \frac{\sum_{j:(i,j)\in E} \beta_{ij}\mathbf{q_j} + \alpha_i \hat{\mathbf{q}}_\mathbf{i}}{\sum_{j:(i,j)\in E} \beta_{ij} + \alpha_i} \tag{1}$$

$\mathbf{q_i}$ is the retrofitted word vector for the core word w_i, $\hat{\mathbf{q}}_\mathbf{i}$ is the aforementioned initial vector for w_i, and α_i is the weight of the initial vector; currently it is set to the number of given feature words w_j for w_i. $\mathbf{q_j}$ is the retrofitted word vector for the given feature word w_j, and β_{ij} is the weight of the given feature word w_j for the core word w_i; currently, the weight β_{ij} is set to 1. Equation (1) multiplies the initial vector $\hat{\mathbf{q}}_\mathbf{i}$ of the core word w_i by the weight α_i, adding the vectors obtained by multiplying the retrofitted vector $\mathbf{q_j}$ of the given feature word w_j by the weight β_{ij} and by dividing it by the sum of both weights. Running the online update for about ten iterations with the retrofitted word vector $\mathbf{q_i}$ as the next initial vector $\hat{\mathbf{q}}_\mathbf{i}$ increases the relationship between each core word and 266 feature words from an average of 9 to an average of 100. The relationship is increased for each core words to expand the feature words given to the core word recursively. The size of the retrofitted word vector is normalized to 1.

Figure 2 presents an example of retrofitting "Disease," which is a feature word and core word in the dictionary. The points of this algorithm are the following.

- The retrofitted word vector is close to the original vector. In the case of "Disease," the original vector is a one-hot vector.
- When the feature words assigned to a retrofitted core word are not expanded as core words, the weights of the feature words are almost equal. When expanded, the weights decrease according to the number of feature words to be expanded.

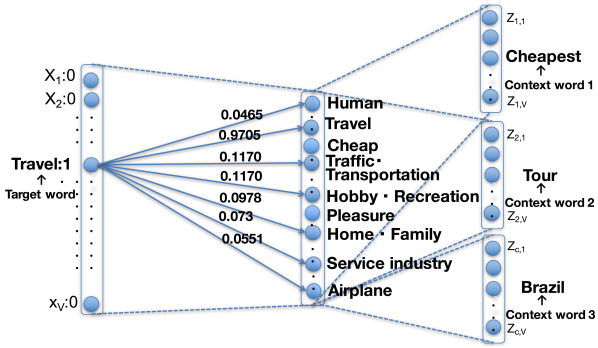

Fig. 3 Skip-gram model setting for testing.

In the vocabulary, each word has two vectors. One is an input vector, which is the weights between the input node and each hidden node, and the other is an output vector, which is the weight between each hidden node and the output node. The retrofitted word vector was used as the seed vector of the input vector. The initial weights of words other than core words were set to 0. Also, the initial weights of the output vector for all words including core words were set to 0, which is the default setting of gensim's doc2vec library[†].

Figure 3 presents an example of the Skip-gram setting for testing the hypothesis. The input layer specifies the target word. The output layer consists of three context words appearing around the target word. The hidden layer comprises the nodes corresponding to 266 feature words. The weights of the target word for each hidden node are retrofitted weights. Each weight is updated by back propagation so that the probability of predicting the context words increases when the target word is input. The objective function is the following [3], [20].

$$E = -\log \sigma\left(\mathbf{v'_w}^T \mathbf{h}\right) - \sum_{w_j \in W_{neg}} \log \sigma\left(-\mathbf{v'_{w_j}}^T \mathbf{h}\right) \tag{2}$$

Because the activation function of the hidden nodes is linear, the hidden layer outputting $\mathbf{h}$ is $\mathbf{v_{w_i}}^T$. $\mathbf{v_w}$ is an input vector with initial weights that are generated by Eq. (1), and $\mathbf{v'_w}$ is the output vector of the word w. W_{neg} is the set of words for negative sampling. The output vector is updated as follows [20].

$$\mathbf{v'_{w_j}}^{(new)} = \mathbf{v'_{w_j}}^{(old)} - \eta\left(\sigma\left(\mathbf{v'_{w_j}}^{(old)T}\mathbf{h}\right) - t_j\right)\mathbf{h} \tag{3}$$

where t_j is 1 when w_j is the context word and 0 otherwise. The initial output vector $\mathbf{v'_w}$ is 0. Thus, the output vectors of the context words become close to the input vector, which is the seed vector, of the target word.

4. Verification of the Hypothesis with a Single Domain Benchmark

In these experiments, we examined the relationship between

[†]https://radimrehurek.com/gensim/models/doc2vec.html

Table 4 Hyper-parameter settings for learning word vectors.

Hyper-parameters	Values
Dimensionality of the feature vectors	266
Number of iterations over the corpus	20
Learning rate	Initial:0.025, Minimum:0.0001
Window size	5
Downsample threshold for words	1e-5
Number of negative sampling words	15

sentiment analysis using a single domain benchmark and readability of tweet embeddings in a user test. We also tested the hypothesis on whether or not weights obtained based on learning and weights based on the dictionary are correlated in a closed test and an open test, compared with a control test.

We used the single domain benchmark of sentiment analysis for Product B and the 560,853 unlabeled tweets in Appendix A. For the 560,853 unlabeled tweets, only noises such as the URL and the account name were deleted. The evaluation benchmark of sentiment analysis consisted of 11,774 tweets of one product brand labeled using crowdsourcing as either positive, negative, or neutral [7]. Japanese morphological analysis, MeCab[†] and its dictionary mecab-ipadic-NEologd[††], which expanded MeCab's default dictionary by millions of new words and named entities from language resources on the Web, were used to extract words from tweets. We used the inflections of verbs and adjectives as different words without transforming to their original forms to let word embeddings learn their context. The number of words extracted from the corpus in five or more times was 30,468, while the number of retrofitted core words was 6,814 words.

4.1 Learning Word Vectors by Our Method and Evaluation of Correlation Coefficients

We updated word vectors using two variants of paragraph vector models with unlabeled tweets only using gensim's doc2vec library. On the basis of the accuracy of the sentiment analysis of the final stage, we decided the values of hyper-parameters for paragraph vector learning of the conventional method. The hyper-parameter settings for learning the corpus are shown in Table 4. Our method used the same hyper-parameter settings. Here, the size of the feature vectors was adjusted to the number of feature words, 266. When the number of dimensions of the feature vectors exceeded 266, we could set the initial value 0 or the random number for the part exceeding 266 in our method. However, no difference occurred in accuracy between 266 dimensions and 300 dimensions for the corpus in the paragraph vector of the conventional method. Thus, we utilized 266 dimensions. Both the PV-DM and PV-DBOW have the same hyper-parameter settings. We used the sum of the input vectors for the hidden layer of the PV-DM for the same reason as with the hyper-parameter settings.

[†]http://mecab.googlecode.com/svn/trunk/mecab/doc/index.html

[††]https://github.com/neologd/mecab-ipadic-neologd

Table 5 Example of retrofitted and learned word vectors for a core word that is a feature word itself.

Generation method	feature words and weights arranged in descending order
Retrofitted vector for "**travel**"	**travel**:0.97, traffic·transportation:0.12, hobby·recreation:0.1, home ·family:0.1, service industry:0.1, airplane:0.06, human:0.05, car:0.05, overseas:0.05, Japan:0.05,
learned vector for "**travel**" by PV-DM	**travel**:1.41, **machine**:0.65, **image**:0.61, company:0.55, state·aspect:0.52, traffic·transportation:0.5, hobby·recreation:0.43, education:0.40, facility:0.38, behavior:0.36,
learned vector for "**travel**" by PV-DBOW	**travel**:1.45, time:0.45, custom:0.44, clothes:0.43, state·aspect:0.43, Europe:0.42, low:0.42, **image**:0.41, public system:0.40, **machine**:0.40,

Table 6 Evaluation results 1: Correlation coefficients between initial and learned word vectors.

	Control Test	Closed Test	Open Test
PV-DM/CBOW	0.224	0.608	0.340
PV-DBOW/Skip-gram	0.211	0.642	0.395

Table 5 shows an example of retrofitted and learned word vectors for a core word "travel," which is a feature word itself. The retrofitted word vector for "travel" was similar to a one-hot vector. The weight of the feature word "travel" of the learned word vector "travel" was more than twice the weight of other feature words in the PV-DM and more than three times the weight of other feature words in the PV-DBOW. Because our method learned the word vectors with a smartphone corpus, "machine" and "image" had higher weights in both of the learned word vectors.

Table 6 presents correlation coefficients between retrofitted vectors and learned vectors in the closed and open test, compared with those of the control test. The control test shows the correlation between the word vectors after learning by the conventional method and the initial vectors, the closed test shows the correlation for the core words used for learning by the proposed method, and the open test shows the correlation for the core words not used for learning by the proposed method as follows.

Control test: We selected the core words (814 words) in the top 2% high-frequency words (2343 words) for the evaluation because high-frequency words had a stronger influence on tweet vector learning than low-frequency words. We evaluated the correlation coefficients between the initial vectors as a control test using default random initialization and the learned vectors.

Closed test: For the 814 core words, we combined all elements of retrofitted word vectors with 0.013 or more as one vector and similarly learned word vectors. A feature word with a value of 0.013 or less corresponded to a relationship according to a two-step recursion with the core word. Therefore, we excluded feature words having a value less than 0.013 from the calculation of the correlation coefficient because the relationship with the core word is not high. Then, we calculated the correlation coefficients of the two vectors. The results showed a stronger correlation compared with that of the control test.

Open test: Let the word vectors learn for the unlabeled tweets excluding the aforementioned 814 retrofitted core word vectors. Subsequently, we calculated the correlation coefficient between the aforementioned 814 retrofitted core

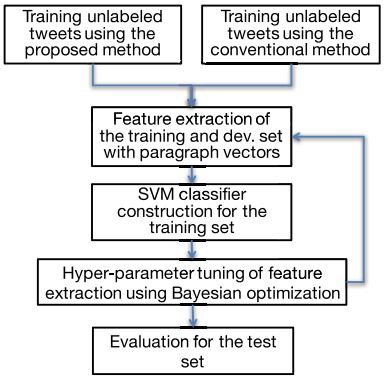

Fig. 4 Procedures of sentiment analysis.

Table 7 Evaluation results 2: Macro-average F-score for predicting positive and negative tweets in 3-class sentiment analysis.

	Dev. Set	Test Set
Conventional Method	68.6	68.8
Our Method	70.5*	70.2**
vs. Conventional Method	*p=0.0006<0.05	**p=1.9e-05<0.05

word vectors with 0.013 or more and the corresponding 814 learned word vectors. The results showed a weak correlation.

4.2 Evaluation of Sentiment Analysis

We evaluated the tasks of sentiment analysis using the paragraph vector of the conventional method and our method. The experimental procedures are presented in Fig. 4. The only difference between the methods was the initial weights when learning the unlabeled tweets. Specifically, the evaluation steps were the following.

[Step 1] In our method, we updated word vectors by having it learn unlabeled tweets with the PV-DM and PV-DBOW based on the retrofitted word vectors, as described in the previous section. In the conventional method, we used the same settings of the hyper-parameters shown in Table 4 for the PV-DM and PV-DBOW based on standard random initialization.

[Step 2] In learning the paragraph vector of the training and dev. set, let the word vectors learned in Step 1 be the initial value of the word vectors. We combined the PV-DBOW and PV-DM for each tweet and created a tweet feature vector. We built a tweet classifier with a support vector machine (SVM) using the labels of each training tweet as training data.

[Step 3] Each tweet vector and its label of the development set was entered into the classifier, and the error rate $[= 100-(F_{pos}+F_{neg})/2]$ was measured. Bayesian optimization automatically adjusted the parameters of the paragraph vector learning so that the output of the objective function, which is the error rate, was minimized [21].

After Step 2 and Step 3 were repeated until the error rate converged, the hyper-parameter of the paragraph vector learning was determined.

As presented in Table 7, we found that the evaluation results of our method were better than those of the conventional method in the macro-average F-score $[(F_{pos}+F_{neg})/2]$ of positive and negative prediction in three-class classification, which is a rating measure utilized in SemEval-2015 task 10 [22]. In this paper, we use the macro-average F-score in percentages as shown in [22].

Table 8 Evaluation results 3: Readability of hidden nodes and macro-average F-score of the corresponding 2-class sentiment analysis.

	Readability of hidden nodes			Sentiment Analysis
Tweet Vectors	Top1	Top5	Top10	F-score
PV-DBOW Positive	64.5%	56.1%	46.6%	86.8
PV-DBOW Negative	66.8%	48.7%	41.5%	78.3
PV-DBOW All	66.1%	52.4%	44.1%	82.5
PV-DM Positive	56.4%	46.2%	36.9%	80.2
PV-DM Negative	67.3%	43.8%	37.8%	74.7
PV-DM All	61.9%	45.0%	37.3%	77.5
Control Test for Positive		16.1%	15.4%	80.2
Control Test for Negative		13.9%	13.9%	74.7
Control Test for All		15.0%	14.6%	77.5

4.3 User Test for Readability

We conducted a readability user test of hidden nodes for the learned tweet vectors. We prepared feature words for hidden nodes with top ten weights for 30 tweets, which were estimated to be positive or negative by using the PV-DM and PV-DBOW individually. The top ten feature words were given for each tweet, and 30 user testers were asked through crowdsourcing whether or not each tweet was associated with the ten feature words. Table 8 shows what percentage of the Top 1, 5, and 10 weighted hidden nodes of PV-DBOW or PV-DM tweet vectors that were classified as either positive or negative were related to the tweets. For the PV-DBOW, the percentage of the given feature words were related to the tweets where one of the paragraph vector models learned the document embeddings was 66.1% for the top weighted hidden node and was 52.4% for the top five weighted hidden nodes. The PV-DM results were 61.9% for the top weighted hidden nodes and 45.0% for the top five weighted hidden nodes. Overall, the PV-DBOW readability was better than that of PV-DM, though the PV-DM results tended to be more readable for negative tweets compared with those of the PV-DBOW. The third results show a control test. The control test assessed how user testers scored on five or ten feature words randomly chosen for the tweets classified as either positive or negative using PV-DM. The reason is that none of the conventional methods which give the meaning of features of tweet embeddings. A comparison with the control test showed that our method apparently improves the readability of distributed representation learning.

Table 8 also shows the macro-average F-score in 2-class sentiment analysis using the corresponding positive and negative vectors for reference. A common trend was evident in the readability of the hidden nodes and sentiment analysis.

For the five cases of the paragraph vector (PV-DBOW) in the aforementioned readability user test, each tweet and a list in descending order of the weight of the feature words

are shown below.

Product B is amazing! (製品 B すげぇ) [power, strong, human, worth, state·aspect, facility, education, positive,]

Product B is a godsend! (製品 B は神) [state·aspect, thought, **human**, relationship, life and death, existence, **power, strong**,]

Oh, after all, the sound of Product B is something good and deep. (あーやっぱり製品 B 音いい w なんか深み？がある w) [sound, advertisement, state·aspect, image, **emotion**, worth, **music**, power, sense,]

While listening to the music with Product B, I was surprised with how good the sound quality was. (製品 B で音楽きいたら音質めちゃくちゃよくてビックリした w …) [state·aspect, **music**, facility, action, ethics, service industry, **emotion**, quantity,]

Even if Product B is fully charged, the LED remains lit. (製品 Bって充電終わっても led 点灯したまんまなんだ…) [brightness, machine, **luminescence**, state·aspect, computer, activity, essence,]

For the first two similar tweets, the common feature words "power," "strong," and "human" had higher weights. Other tweets with the PV-DBOW tweet vectors that were classified as positive and the three feature words with the top ten weights were as follows.

"After all Product B is the strongest. (やっぱ製品 B は最強だから)"

"Because Product B is excellent. (製品 B は優秀だから)"

In the following two tweets on the sound quality of smartphones, the common feature words "music" and "emotion" had higher weights. In the last tweet on charging and LED lighting, the feature words "brightness" and "luminescence" had higher weights. Also, importantly, clear differences emerged in the top feature words of these three groups' tweets.

4.4 Discussion on the Benefits of the Readability

The results of these experiments to test the hypothesis revealed the following benefits for the readability of distributed representations.

- By looking up the top five-ten weighted feature words of the learned words and tweet vectors, you can determine whether or not the learning is proceeding well. You can judge that the learning worked well if about half (40%-60%) of the top five-ten weighted feature words were related to words and tweets. Therefore, the readability can be used for quality assessment of distributed representation learning.
- As shown in Table 8, as the weight is the higher, the readability is the higher. The readability of feature words with the highest weights of tweets is 66.1% on average. These weights represent the strength of the relationship between each feature word and all tweets. Conversely, if you extract the tweet with the highest weight for each feature word in which you are interested, in general, the readability of the feature word for the tweet will be much higher than 66.1%. Thus, you can visualize and extract tweets using the conceptual axis by applying the readability of distributed representations. The conceptual axis can further filter the results of the sentiment analysis for social media mining. Each feature word itself or a combination of feature words becomes a conceptual axis. The image of social media filtering using the conceptual axis is shown in Fig. 5. Tweets can be visualized with a meaningful arrangement along the conceptual axis by constructing the conceptual axis with feature words.

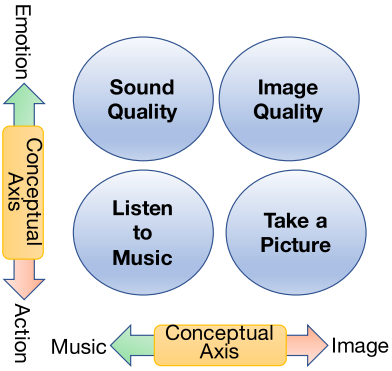

Fig. 5 Application image of social media filtering using the conceptual axis.

5. Expandability Evaluation of the Proposed Method

Because we found support for the effectiveness of our method with a single domain benchmark, we evaluated the expandability of our method with a diverse and large-scale benchmark described in Appendix B. Also, we improved our dictionary for the purpose of testing the expandability of our method. In this section, we describe the improvement of the dictionary and report the evaluation results of the diverse and large-scale benchmark. In addition to the sentiment analysis task, a domain-independent test of our method with a word similarity task was also performed using the Wikipedia corpus.

5.1 The Improvement of the Dictionary

The dictionary aims to increase the readability of word and document embeddings by giving feature words as nodes of the hidden layer of the neural networks. The requirements of the dictionary for this purpose are as follows.

- Because the feature words correspond to each dimension of the paragraph vector, the number of feature words must be a multiple of 4 with more efficient memory alignment.
- In the case of nodes of the hidden layer, the weights of feature words given to many core words increase; therefore, such feature words might be deleted.

Table 9 shows the top feature words given to the core words in Japanese. The feature words "state·aspect" and "relationship·relation" belonging to "abstract concept" in Table 1 are given to many core words. In particular, "state·aspect" is a feature word with larger weights in all

Table 9 Top feature words given to core words

Feature words	Number of core words
state·aspect (様子・様態)	5,188
relationship·relation (関係・関連)	4,460
power·degree (勢力・程度)	3,217
order·regularity (秩序・順序)	2,689
strong (強力)	2,645
positive (肯定的)	2,455

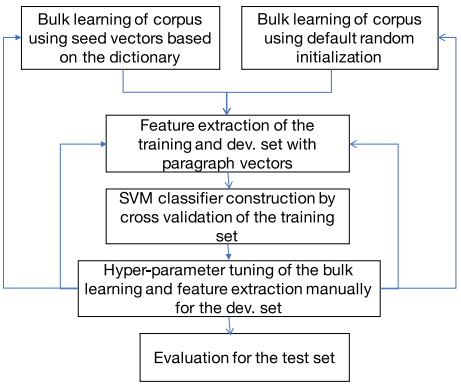

Fig. 6 Procedures of sentiment analysis for the expandability evaluation.

examples shown in Sect. 4.3, which indicates that the influence of the feature word is too strong. Feature words below those shown in Table 9, when used for word expansion, play an important role in estimating positiveness [7]. Therefore, "state·aspect" and "relationship·relation" were deleted, and the number of feature words was set to 264 kinds, which is a multiples of 4.

5.2 Evaluation with the Diverse and Large-Scale Benchmark

We made seed vectors of Japanese core words using the dictionary and conducted an evaluation on 2-class classifications of the diverse and large-scale benchmark in Appendix B using two types of paragraph vector models. The evaluation flow of sentiment analysis using the benchmark is shown in Fig. 6. The differences from the procedure shown in Fig. 4 are that the hyper-parameter tuning was done manually without using Bayesian optimization and that hyper-parameters of bulk learning including seed vectors construction from the dictionary were also adjusted based on the macro-average F-score of the development set. We divided the corpus for bulk learning and the corpus for feature extraction and cross-validation as follows.

Corpus for bulk learning: Total 3.1 M lines (Others: 2.2M tweets, Unlabeled (in the single domain benchmark): 0.56M tweets, Training set: 0.34M tweets, The dictionary: 20K core words and their feature words)

Corpus for feature extraction and cross-validation: 2 class training set: 25,718 tweets, 2class dev. set: 6429 tweets, 2class test set: 6429 tweets

The hyper-parameter settings are shown in Table 10. The seed vectors of the output vectors were also set from the dictionary in this evaluation because the macro-average F-score of the dev. set was improved. The number of iterations in the dictionary was one, so the seed vectors of the output vectors were non-symmetric with the input vectors. We utilized the sum of the input nodes regarding the hidden layer of PV-DM because of the better macro-average F-score of the dev. set.

Table 10 Hyper-parameter settings for learning word and paragraph vectors.

Hyper-parameters	PV-DM	PV-DBOW
# of iterations in the dictionary (input vector)	10	10
# of iterations in the dictionary (output vector)	1	1
# of iterations for bulk learning	20	60
# of iterations for feature extraction	200	60
Word appearance frequency threshold	10	3
Window size (Conventional Method)	2 (5)	5 (5)
Downsample threshold for words	1e-6	1e-5

Table 11 Evaluation results 4: Correlation coefficients between seed vectors and learned word vectors.

	PV-DM		PV-DBOW	
	Input Vector	Output Vector	Input Vector	Output Vector
After bulk learning	0.541	0.682	0.494	0.526
After feature extraction	0.423	0.686	0.494	0.526

Table 12 Evaluation results 5: Macro-average F-score for predicting positive and negative tweets in 2-class sentiment analysis.

	PV-DM		PV-DBOW	
	Conventional Method	Our Method	Conventional Method	Our Method
Dev. set	86.1	87.1*	88.85	89.01**
Test set	85.4	86.5***	88.02	88.19****
vs. Conventional Method	*p=0.0019<0.05, ***p=0.0072<0.05		**p=0.17>0.05, ****p=0.13>0.05	

Table 11 presents the correlation coefficients between seed vectors and learned vectors in the closed test after bulk learning of the corpus and after feature extraction of the training and dev set. As for PV-DM, as shown in Fig. 1, because the tweet ID was added to the context words, the word vectors were also updated when the feature extraction of tweets was done. However, as for PV-DBOW, the update of word vectors was stopped when the feature extraction of tweets was done because the extraction and the update of word vectors are independent. Therefore, the correlation of the word vectors of PV-DBOW is lower than that of the single domain benchmark shown in Table 6, but they still correlate with the seed vectors. With regard to PV-DM, the correlation of input vectors decreased as the scale of benchmarks increased, but it was considerably higher than the correlation of control tests shown in Table 6, and the correlation of output vectors was maintained. Therefore, in both the PV-DBOW and PV-DM, the meaning of the nodes in the hidden layer was maintained.

Table 12 presents the macro-average F-score in 2-class sentiment analysis. A comparison of the results of the single domain small benchmark shown in Table 8 revealed that the F-score improved by 9 points in PV-DM and by 5.69 points in PV-DBOW in the new benchmark. In the evaluation of the new benchmark, our method is still better than the conventional method in the PV-DM, and it is comparable to the conventional method in the PV-DBOW.

Also, as shown in the following examples, the top five to ten weighted feature words were found to be related

Table 13 Relationship between the correlation coefficients after feature extraction and the task accuracy of the dev. set for the PV-DM.

Number of Bulk Learning	iter=0	iter=1	iter=5	iter=10	iter=15	iter=20
Macro Average F-score	84.5	84.7	85.9	86.7	86.8	87.1
Correlation Coefficients (Input Vector)	0.845	0.831	0.751	0.650	0.534	0.423

Table 14 Relationship between the correlation coefficients after feature extraction and the task accuracy of the dev. set for the PV-DBOW.

Number of Bulk Learning	iter=0	iter=1	iter=2	iter=10	iter=40	iter=60
Macro Average F-score	86.06	87.90	88.44	88.69	88.80	89.01
Correlation Coefficients (Input Vector)	1.0	0.598	0.571	0.502	0.500	0.494

to tweets of diverse categories concerning robot cleaners, printing services, and makers also having higher weights.

Robot cleaner B, Wonderful! The air in the room got really clean. (ロボット掃除機 **B** すげーわ部屋の空気がすごいきれいになった) [**dwelling**, family, **facility, environment, structure**, newness, **machine**,]

I am feeling the Maker A's love for robot cleaner A and ... (**A** 社さんのロボット掃除機 **A** と政宗様への愛をひしひしと**...**) [**customer, kindness**, **dwelling**, art, **peace**, sound, **company**, human,]

It is really useful to create booklets with the printing service at convenience stores. (コンビニプリントで小冊子作れるのほんと便利) [**book**, power, **worth**, newness, **manufacture**, education,]

Oh dear! Maker B is allowed to marry same-sex within the company? It is wonderful that companies promote it! (え！**B** 社社内同性婚 **ok**！？企業がすすめるってすごい!!) [**company**, commerce, worth, **public_system**, economy, international_relation, **social_activity**,]

Next, we examined the relationship between readability, which here is the correlation coefficients between seed vectors and learned word vectors, and the task accuracy. Table 13 shows the relationship between the correlation coefficients after bulk learning and feature extraction, and the task accuracy of the dev. set, which is the macro average F-score, for the PV-DM. Also, Table 14 shows it for the PV-DBOW.

In the PV-DM, which learns the word order information, the readability monotonously decreases according to the number of bulk learning of 3.1 M corpus, and the task accuracy peaks at 20 iterations. A trade-off relationship exists between readability and task accuracy up to the 20 iterations.

In the PV-DBOW, which learns the context information, the learning is rapidly performed by two iterations of bulk learning. A trade-off relationship also exists between readability and task accuracy, up to two iterations, but after that, the change in readability and task accuracy is small until 60 iterations. The "iter = 0" in Table 14 is the case where feature extraction of tweets was performed only using the seed vectors of the core words based on the dictionary, and the F-score of the sentiment analysis was 86.06. By adapting the word vectors to the domain using about 3.1 M corpus, the F-score has increased to 89.01.

The realistic number of bulk learning is ten iterations for the PV-DM and two iterations for the PV-DBOW because both the readability and task accuracy are balanced.

5.3 Domain Independent Test

The original word semantic vector dictionary was developed to embed the knowledge of the encyclopedia [6]. Recently, Wikipedia[†], which is a free encyclopedia, provides a Japanese domain independent corpus. Also, the performance of word embeddings is usually evaluated in a word similarity task. Moreover, the first word-similarity dataset in Japanese has been published [23]. The dataset consists of four parts of speech, i.e., adjectives, adverbs, verbs, and nouns, and it contains more than 4500 word pairs including rare words in addition to common words. We evaluated the word similarity task using the dataset for word embeddings learned with the Wikipedia corpus. In addition, the words of each part of speech were randomly chosen to show feature words with the top n weights and examples of the top n synonyms using word embeddings of our method.

Table 15 Statistical information on the learning of the Wikipedia corpus

Items	Values
Number of paragraphs	1.19 M articles
Number of uniq words	0.55M (more than 10 articles appeared)
Size of word corpus	366.23M (more than 10 articles appeared)
Number of core words	15,866 (more than 10 articles appeared)
Required memory	2.94G
Required time for 3 iters	44 min (46 min) (Our Method (266 features)), 39 min (Conventional Method)

A word similarity task evaluated word embeddings constructed by our method and the conventional method both learning the Wikipedia corpus using the PV-DBOW/Skip-gram model. In the Japanese word similarity dataset[††], each pair of words had an average point of similarity given by ten human annotators. The similarity between word pairs by each method was estimated by the cosine measure of the word vectors. If multiple words with word vectors were extracted from one word in the dataset by the Japanese morphological analysis, MeCab and its dictionary mecab-ipadic-NEologd, the weighted sum of the vector of each word was calculated. The weights of the constituent words were set to the reciprocal of appearance order for adverbs and verbs, and the constituent words were set to have equal weights for adjectives and nouns. Also, all constituent words must have word vectors. The performance was evaluated by calculating the Spearman rank correlation coefficient between the word similarity in the Japanese word similarity dataset and the word similarity estimated by each method.

We removed all tags from the Wikipedia corpus[†††] and modified it to one article per line. The experiments on the relationship between the readability of the learned word vectors and the task accuracies revealed that the number of realistic bulk learning is two iterations for the PV-DBOW, as shown at the end of the previous section. We confirmed that Epoch 3 gives the best performance in the word similarity task in both the conventional method and the proposed method. So the number of learning iterations over the corpus in PV-DBOW was set to three (Epoch 3). Statistical information on the learning of the Wikipedia corpus is shown in Table 15. By changing 266 feature words to 264 feature words, the learning time improved by about 2 minutes. The

[†]https://en.wikipedia.org/wiki/Wikipedia
[††]https://github.com/tmu-nlp/JapaneseWordSimilarityDataset
[†††]https://dumps.wikimedia.org/jawiki/latest/

difference in the learning time with the conventional method is mainly due to an increase in file IO.

The correlation coefficients shown in Table 16 were for a closed test between the seed vectors based on the core words of 15,866 words appearing in more than ten articles in the Wikipedia corpus and the word vectors learned with the Wikipedia corpus. Because the size of the word corpus of the Wikipedia corpus is about five times that of the Twitter corpus, the correlation coefficient decreased, but the learned word vector was still correlated with the seed vector.

Table 17 presents the evaluation results on the word similarity task using the Wikipedia corpus. The evaluation results of our method were comparable to those of the conventional method in the Spearman rank correlation coefficient.

Both of the evaluation results showed that the core words learned with the Wikipedia corpus are correlated with the seed vectors based on the dictionary and that the evaluation results of our method are comparable to those of the conventional method in the word similarity task. Table 18 shows examples of the top n weighted feature words and the top n similar words for core words of each part of speech randomly selected. The boldface feature words are feature words that are given to the core words in the dictionary and that remained after learning the Wikipedia corpus. For example, the feature word of a special word such as "education" had a very high weight in a technical term such as "degree." For the top five weighted feature words of an adverb such as "incomprehensible," which is a rare word, three feature words remained from the dictionary, and two reasonable feature words were learned from the context. For a verb such as "get drunk," only one feature word "food" remained, but others were appropriate feature words given in learning from the context. Top n similar words based on word vectors are appropriate in any of the aforementioned cases. Therefore, our method is domain-independent.

Table 16 Evaluation results 6: Correlation coefficients (closed test)

	Input Vectors	Output Vectors
PV-DBOW/Skip-gram	0.432	0.476

Table 17 Evaluation results 7: Word similarity task using Wikipedia corpus.

Part of Speech	Conventional Method	Our Method
Adjective	0.3470	0.3334
Adverb	0.2311	0.2415
Verb	0.3323	0.3431
Noun	0.2888	0.2840
Macro Average	0.2998	0.3005*

vs. Conventional Method *p=0.91>0.05

6. Conclusion

We proposed a new method to give specific meaning to each node of a hidden layer in neural networks using a word semantic vector dictionary to enable the readability of word and document embeddings.

First, using a single domain sentiment analysis benchmark, we found that the evaluation results of our method were better than those of the conventional method in the macro-average F-score. Also, we tested the readability of tweet embeddings in a user test. A total of 52.4% of the top five weighted feature words were related to tweets.

Next, we improved the dictionary, which is suitable for distributed representation, and constructed a large-scale sentiment analysis benchmark consisting of eight categories to evaluate the expandability of our method. In the evaluation of the benchmark, we found that our method is still better than the conventional method in the PV-DM and that it is comparable to the conventional method in the PV-DBOW. Compared to the results of the single domain benchmark, the F-score improved by 9 points in PV-DM and by 5.69 points in PV-DBOW. Also, we found that the top five to ten weighted feature words were related to tweets of diverse categories.

Finally, word similarity tasks using the Wikipedia corpus were evaluated. The evaluation results of our method were comparable to those of the conventional method in the Spearman rank correlation coefficient. Moreover, we found the top five weighted feature words to be related to the core words.

Also, our experimental results demonstrated that weights obtained based on learning and weights based on the dictionary are more strongly correlated in a closed test and more weakly correlated in an open test, compared with the results of a control test. As the word corpus used for learning expanded, the correlation with the seed vector decreased, but the correlation was maintained as high as about twice that of the default random setting.

The proposed method improved the readability of distributed representations because these distributed representations of words and paragraphs learned by neural networks are weights for each hidden node with a specific meaning.

Table 18 Example of Top n weighted feature words and similar words for core words

Core words	excellent (素晴らしい)	incomprehensible (不可思議)	get drunk (酔う)	degree (学位)
Top n weighted feature words	**worth**: 0.658 **positive**: 0.517 power-degree: 0.499 machine: 0.473 physics: 0.341	**idea**: 0.729 image: 0.679 **religion**: 0.534 **phenomena**: 0.450 sense: 0.427	traffic-transportation: 0.686 emotion: 0.642 **food**: 0.644 idea: 0.562 power-degree: 0.484	**education**: 1.763 **public_system**: 0.777 **special**: 0.727 possess: 0.668 physics: 0.548
Top n Similar Words based on the word vectors	素晴らしく: 0.757 すばらしい: 0.754 素晴らしかっ: 0.744 すばらしく: 0.667 (thus far readings or inflection forms) perfect: 0.640	strange: 0.685 mystery: 0.633 eerie (不気味): 0.622 unusual:0.617 knowledge (人智): 0.616 Bilocation (超常現象): 0.607	drunk: 0.733 drunken sickness: 0.705 gloom (憂さ): 0.703 excessive drinking: 0.702 heavy drinking: 0.695 merry drinker: 0.686	master: 0.849 doctor's degree: 0.827 bachelor's degree: 0.806 master's degree:0.803 ph.d.: 0.793 bachelor: 0.784

Future research will be conducted to optimize 264 feature words. Learning the Wikipedia corpus with our method and displaying the top n weighted Wikipedia articles of each feature word will enable clarifying important feature words and trivial feature words in user tests. We will also determine whether or not our method is universal and applicable to other languages using a standard English dataset of a sentiment analysis task, word similarity task, and word analogy task (see Appendix C). We can also evaluate related work [11], [13] using WordNet as a regularizer of the learning process in our method. We believe that the performance of the task of social media mining could be improved by our method without any annotated data because the performance of sentiment analysis and the readability of document embeddings show similar trends.

Acknowledgments

The research results have been achieved by "Research and Development on Fundamental and Utilization Technologies for Social Big Data" the Commissioned Research of National Institute of Information and Communications Technology (NICT) and NAIST Bigdata Project. This study was partially supported by CREST JST. It was also supported by Sharp Corporation.

References

[1] I. Keshi, Y. Suzuki, K. Yoshino, and S. Nakamura, "Semantically readable distributed representation learning for social media mining," Proc. IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp.716–722, 2017.
[2] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," CoRR, vol.abs/1301.3781, 2013.
[3] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," CoRR, vol.abs/1310.4546, 2013.
[4] T. Mikolov, W. Yih, and G. Zweig, "Linguistic regularities in continuous space word representations," Proc. NAACL, pp.746–751, 2013.
[5] Q.V. Le and T. Mikolov, "Distributed representations of sentences and documents," Proc. ICML, pp.1188–1196, 2014.
[6] I. Keshi, H. Ikeuchi, and K. Kuromusha, "Associative image retrieval using knowledge in encyclopedia text," Systems and Computers in Japan, vol.27, no.12, pp.53–62, 1996.
[7] I. Keshi, Y. Suzuki, K. Yoshino, G. Neubig, K. Ohara, T. Mukai, and S. Nakamura, "Reputation information extraction from twitter using a word semantic vector dictionary," IEICE Trans. Inf. & Syst. (Japanese Edition), vol.J100-D, no.4, pp.530–543, April 2017.
[8] G.A. Miller, "WordNet: A lexical database for English," Commun. ACM, vol.38, no.11, pp.39–41, Nov. 1995.
[9] C.F. Baker, C.J. Fillmore, and J.B. Lowe, "The Berkeley FrameNet project," Proc. ACL-COLING, pp.86–90, 1998.
[10] S. Vosoughi, P. Vijayaraghavan, and D. Roy, "Tweet2vec: Learning tweet embeddings using character-level cnn-lstm encoder-decoder," Proc. SIGIR, pp.1041–1044, 2016.
[11] C. Xu, Y. Bai, J. Bian, B. Gao, G. Wang, X. Liu, and T.-Y. Liu, "RC-NET: A general framework for incorporating knowledge into word representations," Proc. CIKM, pp.1219–1228, 2014.
[12] J. Pennington, R. Socher, and C.D. Manning, "Glove: Global vectors for word representation," Proc. EMNLP, pp.1532–1543, 2014.
[13] D. Bollegala, A. Mohammed, T. Maehara, and K.i. Kawarabayashi, "Joint word representation learning using a corpus and a semantic lexicon," Proc. AAAI, pp.2690–2696, 2016.
[14] M. Faruqui, J. Dodge, S.K. Jauhar, C. Dyer, E. Hovy, and N.A. Smith, "Retrofitting word vectors to semantic lexicons," Proc. NAACL-HLT, pp.1606–1615, May–June 2015.
[15] D.M. Blei, A.Y. Ng, and M.I. Jordan, "Latent dirichlet allocation," J. Mach. Learn. Res., vol.3, pp.993–1022, March 2003.
[16] Y. Liu, Z. Liu, T.S. Chua, and M. Sun, "Topical word embeddings," Proc. AAAI, pp.2418–2424, 2015.
[17] Q. Li, S. Shah, X. Liu, A. Nourbakhsh, and R. Fang, "Tweet topic classification using distributed language representations," Proc. IEEE/WIC/ACM International Conference on Web Intelligence (WI), pp.81–88, 2016.
[18] J.H. Lau and T. Baldwin, "An empirical evaluation of doc2vec with practical insights into document embedding generation," Proc. Workshop on Representation Learning for NLP, pp.78–86, 2016.
[19] S. Suzuki, "Probabilistic word vector and similarity based on dictionaries," Proc. CICLing, pp.564–574, 2003.
[20] X. Rong, "word2vec parameter learning explained," CoRR, vol.abs/1411.2738, 2014.
[21] J. Snoek, H. Larochelle, and R.P. Adams, "Practical bayesian optimization of machine learning algorithms," Advances in Neural Information Processing Systems, pp.2951–2959, 2012.
[22] S. Rosenthal, P. Nakov, S. Kiritchenko, S.M. Mohammad, A. Ritter, and V. Stoyanov, "SemEval-2015 Task 10: Sentiment analysis in Twitter," Proc. SemEval-2015, pp.451–463, 2015.
[23] Y. Sakaizawa and M. Komachi, "Construction of a japanese word similarity dataset," CoRR, vol.abs/1703.05916, 2017.
[24] M.P. et al., "SemEval-2016 Task 5: Aspect Based Sentiment Analysis," Proc. SemEval-2016, pp.19–30, 2016.

Appendix A: Single Domain Benchmark for Japanese Twitter Sentiment Analysis

We made a single domain benchmark using crowdsourcing to label each tweet for each product brand [7]. There are four kinds of labels as follows.

Positive: Positive opinions are stated for the target product brand in a tweet.

Negative: Negative opinions are stated for the target product brand in a tweet.

Neutral: Opinions on the target product brand in a tweet are neither positive nor negative.

Unrelated: Opinions on the target product brand are not stated.

We allocated five workers to each tweet and assigned labels for approximate 35,000 tweets. Table A· 1 shows the configuration of benchmarks constructed using crowdsourcing. Tweets with the same number of votes on multiple la-

Table A· 1 Configuration of the benchmark.

Dataset	Positive	Negative	Neutral	Total	Unrelated
Product A					
Training set	1122	1023	1065	3210	
Dev. set	280	256	266	802	
Test set	280	256	266	802	
Total number	1682	1535	1597	4814	8216
Product B					
Training set	3654	2375	2802	8831	
Dev. set	608	396	467	1471	
Test set	609	396	467	1472	
Total number	4871	3167	3736	11,774	5998
Unlabeled tweets					
560,853					

Table A·2 Number of labels given to tweets by crowdsourcing

Categories	Number of tweets	Positive	Negative	Neutral	Positive&Negative	Unrelated	Number of labels
Smartphone A	130,650	2,906	5,188	16,054	594	68,158	92,900
Smartphone B	482,036	5,655	9,531	51,900	603	18,884	86,573
Smartphone C	1,155,034	3,543	6,176	45,568	408	28,844	84,539
Robot cleaner A	11,664	741	311	6,894	41	4,371	12,358
Robot cleaner B	307,156	954	1,089	20,654	55	48,092	70,844
Printing service	275,097	3,887	3,484	30,176	241	35,514	73,302
Maker A	187,584	744	4,421	40,950	75	26,358	72,548
Maker B	169,532	1,503	937	13,624	80	54,891	71,035
Total number	2,718,753	19,933	31,137	225,820	2,097	285,112	564,099

Table A·3 Japanese Twitter Sentiment Analysis Benchmark

Dataset	Positive	Negative	2 class total	Neutral	Unrelated	Total
Training set	10,100	15,618	25,718	137,089	180,186	342,993
Dev. set	2,525	3,904	6,429	34,272	45,046	85,747
Test set	2,525	3,904	6,429	34,272	45,046	85,747
Total number	15,150	23,426	38,576	205,633	270,278	514,487
Others			2,204,266			

Table A·4 Specifications of the dictionary

	number of core words	Average feature word number
Japanese	20,330 words	8.77 feature words
English	21,912 words/phrases	11.73 feature words

audience hobby sport enjoyment majority general individual art image music,
audience_seats sport structure facility space art image music customer
audio human language sound hobby machine music electronics
audio_source machine computer software sound
audio_training japan knowledge discussion book literature language
audiovisual human_body visceral_organ health sense medicine
audiovisual_senses human_body visceral_organ health sense medicine

Fig. A·1 Example of the dictionary

bels in the 1st place and the "unrelated" labels were removed from the experiments in Sect. 4.

Appendix B: Diverse and Large-Scale Benchmark

We constructed a large-scale benchmark with diversity using crowdsourcing to test the expandability of our method. Table A·2 shows the number of tweets collected for each category, which consists of five product brands, one service and two makers, and the number of tweets labeled by crowdsourcing. We collected tweets with keywords such as product brands for 13 months from October 2014 to November 2015 regarding smartphones A and B and for 13 months from January 2015 to February 2016 regarding other categories. We labeled the tweets of each category using crowdsourcing. Five workers were assigned to each tweet, and labels were given by majority vote. In the case of ties in the majority vote, multiple labels were assigned to one tweet. Five kinds of labels were used, as follows.

Positive: Positive opinions are stated for specific features of a target category in a tweet.
Negative: Negative opinions are stated for specific features of a target category in a tweet.
Neutral: Opinions on a target category in a tweet are neither positive nor negative in the sense of the aforementioned.
Positive&Negative: Both positive and negative opinions are stated in a tweet.
Unrelated: Opinions on a target category are not stated.

Here, the point was to clarify positive and negative judgment criteria as "specific features" for target categories, like aspect-based sentiment analysis [24], to perform crowdsourcing work efficiently on large-scale tweets. Therefore, the first two examples of Sect. 4.4, which have been usually considered positive, became neutral in the labeling this time. Also, the latter three cases that express their opinions on specific features "sound" and "charging" are subjects of positive and negative judgment.

Table A·3 shows the Japanese Twitter sentiment analysis benchmark created based on the result of labeling by crowdsourcing. We classified tweets that are given multiple labels including "positive & negative" labels in the "Others." The 2-class classifications of positive and negative were part of a diverse and large-scale benchmark of 38,576 tweets, even compared with the benchmark of SemEval [22]. In addition, the reliability of the benchmark was high because the positive and negative classification standards were clear. Furthermore, because the specific features of products, services, and organizations were stated in the tweet, the test was considered to be a benchmark to extract tweets that are useful for product planning and quality support.

Appendix C: English Version of the Dictionary

Regarding English version construction, first, we translated 20,330 core words from Japanese to English words or English phrases using the neural machine translation API[†]. The reason for using neural machine translation rather than a Japanese-English dictionary was to translate difficult words extracted from encyclopedia or newspapers into English words and phrases used every day. As a result, they were translated into about 14,000 unique English words/phrases including translation errors. We employed three proofreaders using crowdsourcing to make the proofreading request of translated English words and phrases while referring to given feature words. The specifications of the dictionary Ver. 2 are shown in Table A·4. Figure 5 shows an example of the dictionary. The core word is placed at the beginning of each line, and the feature words are listed with a space delimited. An English phrase is expressed by combining English words and underscores. The problem with the English version is that because we only merged the results of three proofreaders, it contains translation errors including misspellings, and English core words translated from various Japanese core words are given all the original feature

[†]https://www.microsoft.com/en-us/translator/

words. For example, in Fig. 5, the feature words of the core word "audio" merge the original feature words of the core words "voice" and "audio." That is why the average feature number of English words/phrases is larger than that of Japanese, as shown in Table A· 4. Also, "audio_training" is a translation error of "kanji_reading," and "audiovisual_sense" is better than "audio_visual" for given feature words. The English version is planned to be improved after publication.

Ikuo Keshi received his B.E. and M.E. degrees in Electronic Engineering from Osaka University and Ph.D. degree from Nara Institute of Science and Technology in 1981, 1983 and 2017. In 1983, he joined Sharp Corporation. In 1989, he had been a visiting Scientist at the AI Lab of MIT (U.S.) and had engaged in the R&D of AI, home healthcare and big data as a chief technical research fellow. He left Sharp in Sept., 2015. He is currently a professor in the Department of Electrical and Electronic Engineering, Fukui University of Technology. He is a member of IEICE, IPSJ and JSAI.

Yu Suzuki rceived his M.E. and Ph.D. degrees from Nara Institute of Science and Technology in 2001 and 2004. He became an assistant professor at Ritsumeikan University in 2004, a researcher at Kyoto University in 2009, and an assistant professor at Nagoya University in 2010. He is currently an associate professor at Nara Institute of Science and Technology. He is a member of IPSJ, IEICE, DBSJ, IEEE-CS, and ACM.

Koichiro Yoshino received his B.A. degree in 2009 from Keio University, M.S. degree in informatics in 2011, and Ph.D. degree in informatics in 2014 from Kyoto University. From 2014 to 2015, he was a Research Fellow (PD) of Japan Society for Promotion of Science. Currently, he is an assistant professor at Nara Institute of Science and Technology. Dr. Koichiro Yoshino received the JSAI SIG-research award in 2013. He is a member of IEEE, ACL, IPSJ, and ANLP.

Satoshi Nakamura received his B.S. from Kyoto Institute of Technology in 1981 and Ph.D. from Kyoto University in 1992. He was a director of ATR in 2000-2008, and a vice president of ATR in 2007-2008. He was a director general of Keihanna Research Laboratories, National Institute of Information and Communications Technology, Japan in 2009-2010. He is currently a professor and a director of Augmented Human Communication laboratory at Nara Institute of Science and Technology.