PAPER

Utterance Intent Classification for Spoken Dialogue System with Data-Driven Untying of Recursive Autoencoders

Tsuneo KATO[†a)], ***Member***, **Atsushi NAGAI**[†], ***Nonmember***, **Naoki NODA**[†], **Jianming WU**[††], ***Members***, ***and*** **Seiichi YAMAMOTO**[†], ***Fellow***

SUMMARY Data-driven untying of a recursive autoencoder (RAE) is proposed for utterance intent classification for spoken dialogue systems. Although an RAE expresses a nonlinear operation on two neighboring child nodes in a parse tree in the application of spoken language understanding (SLU) of spoken dialogue systems, the nonlinear operation is considered to be intrinsically different depending on the types of child nodes. To reduce the gap between the single nonlinear operation of an RAE and intrinsically different operations depending on the node types, a data-driven untying of autoencoders using part-of-speech (PoS) tags at leaf nodes is proposed. When using the proposed method, the experimental results on two corpora: ATIS English data set and Japanese data set of a smartphone-based spoken dialogue system showed improved accuracies compared to when using the tied RAE, as well as a reasonable difference in untying between two languages.

key words: *utterance intent classification, spoken dialogue system, recursive autoencoder, data-driven untying*

1. Introduction

In accordance with the recent popularity of spoken dialogue systems such as smart speakers, spoken dialogue technology is anticipated to become more sophisticated. Spoken dialogue systems are expected to correctly recognize the topic and intention of a user's utterance, and give an appropriate response. Spoken language understanding (SLU) is an essential component of a spoken dialogue system. SLU has to correctly estimate the intent and topic of a user's utterance despite of the variety of oral expressions used. A basic approach has been to classify the output text of automatic speech recognition (ASR) into one of a predefined set of intent classes, followed by slot filling specific to the estimated intent class.

Traditionally, the classification of the user's utterance intent was made based on a bag-of-words representation or its extension to N-grams of the words. The criterion was maximizing the cosine distance to the bag-of-words representations of the predefined intent classes or with classifiers such as an SVM [1], [2] and maximum-entropy classifier [3]. In such bag-of-words systems, the relations between words were given by a thesaurus such as WordNet [4]. However, thesauri have problems such as expensive in development and maintenance, hard to adapt meaning changes over time and difficult to express difference of nuances.

In contrast, continuous vector models of words such as Word2Vec [5] and GloVe [6] were successful in capturing meaning of words. These models learn an embedding of words into a real vector space of a relatively low dimension by estimating likely words conditioned with their contexts using a large text corpus. As it is not straightforward to obtain embeddings for longer phrases and sentences, there have been various compositionality technique proposals that estimate real vectors for phrases and clauses through arithmetic operations on word embeddings. Neural network techniques have been applied to the nonlinear arithmetic operations for the compositionality.

Recurrent neural network (RNN) models with Long Short-Term Memory (LSTM) and attention mechanism are actively studied as neural network models capturing semantics from time series data. In SLU tasks, RNN models accept the sequence of word indices in the order of time and/or in the reverse order, and the word indices are converted to embeddings internally. Luan et al. [7] reported that RNN-based SLU was improved by pre-training with word embeddings. Liu and Lane proposed joint SLU techniques that estimate utterance intention, slot filling and next word prediction [8] and that estimate utterance intent and slot filling with attention-based RNN [9]. Chen et al. tried to incorporate syntactic or semantic structures of sentences as knowledge-guided structural attention networks into a RNN, which is structured as a linear chain temporally [10].

In contrast to RNN models, the recursive neural network models accept a word sequence, but have the latitude of coupling a node with either its preceding or succeeding node. This mechanism allows the neural network based compositionality technique to incorporate syntactic parsing. It has a potential to achieve a balance between soft compositionality of word embeddings and hard syntactic parsing. Japanese, an agglutinative language, has a relatively flexible word order though it has an underlying subject-object-verb order. In colloquial expression, the word order becomes more flexible. We think the recursive neural network models are suitable for SLU of Japanese colloquial expressions. Socher et al. showed promising results in polarity estimation and sentiment distribution estimation with a recursive autoencoder (RAE) [11]. However, the recursive neural network models utilized little syntac-

Manuscript received September 20, 2018.
Manuscript revised January 14, 2019.
Manuscript publicized March 4, 2019.

[†] The authors are with the Graduate School of Science and Engineering, Doshisha University, Kyotanabe-shi, 610–0394 Japan.
[††] The author is with KDDI Research, Inc., Fujimino-shi, 356–8502 Japan.
a) E-mail: tsukato@mail.doshisha.ac.jp

DOI: 10.1587/transinf.2018EDP7319

tic information. As a noticeable example, a single autoencoder was applied to all nodes in a tree in the applications of RAE. However, the arithmetic operation between nodes is intrinsically different depending on the combination of child nodes, and it is difficult to represent this operation with only a single autoencoder. To solve this problem, Socher proposed explicit word-dependent operations in a matrix-vector model [12]. This model had to estimate a huge number of parameters. In his next proposal of Compositional Vector Grammars (CVGs) [13], which combined Probabilistic Context-Free Grammar (PCFG) with compositional vector models, recursive neural networks were untied by the types of child nodes. Hermann and Blunsom incorporated Combinatory Categorial Grammar (CCG) [14] into an RAE [15]. Guo et al. proposed joint utterance intent classification and slot filling with syntactic type dependent recursive neural networks [16]. However, the number of syntactically untied neural networks is likely to be excessive and it is difficult to define appropriate syntactic untying manually in practice. The estimation of model parameters can readily fall into the data sparseness problem.

Hence, we propose a data-driven untying of autoencoders based on a regression tree with part-of-speech (PoS) information to obtain an efficient untying of the recursive autoencoder (RAE). The regression tree is formed with predictor parameters of PoS tags of the left and right child nodes to reduce the total of an error function. We evaluate the proposed method with English ATIS corpus and Japanese corpus of a smartphone-based spoken dialogue system [17] to see generality and difference of the methods between two languages. We compare the accuracies of utterance intent classification between the RAEs of a single tied autoencoder, autoencoders untied by a manually defined rule, and autoencoders untied by the data-driven splitting.

The remainder of the paper is structured as follows. In Sect. 2, Socher's RAE as the baseline method and some previous studies incorporating syntactic information into recursive neural network models are introduced. In Sect. 3, two types of English and Japanese data used for utterance intent classification are described. In Sect. 4, the basics of RAE, rule-based syntactic untying and data-driven untying of RAE are explained. Experiments of utterance intent classification and the regression tree generated by data-driven untying are discussed in Sect. 5. Finally, the conclusion is given in Sect. 6.

2. Related Studies

Socher et al. applied an autoencoder repeatedly to a word sequence to obtain a sentence-level vector representation and to estimate distribution of five sentiment labels [11]. The RAE can reflect a hierarchical structure of a sentence on compositionality of word vectors. The training of the RAE and the estimation of an utterance intent class with the RAE are explained in Sect. 4.1.

The next proposal by Socher et al., Recursive Matrix-Vector neural network [12], modeled the inherent meaning with the vector and how it changes the meaning of neighboring words or phrases with the matrix explicitly. The model showed a promising result in classifying semantic relationship such as cause-effect or topic-message between nouns. However, the model had to estimate a huge number of parameters.

Compositional Vector Grammar (CVG) proposed by Socher et al. [13] was a combination of Probabilistic Context Free Grammar (PCFG) with syntactically untied recursive neural networks. Vectors of non-leaf nodes were computed by a recursive neural network which was conditioned on syntactic categories from a PCFG. The weights of the neural network were dependent on the categories of the child nodes. CVG improved its parsing accuracy on WSJ section 23 from 86.6% to 90.4% over the Stanford Parser.

Hermann and Blunsom incorporated syntax into RAE in combination with Combinatory Categorial Grammar (CCG). Their Combinatory Categorial Autoencoders (CCAE) [15] switched a set of nonlinear arithmetic operations for compositionality at any point in a parse tree based on the CCG formalism. The model was more compact than the Recursive Matrix-Vector model due to the efficient combinators of CCG. They trained several CCAE models making increasing use of the CCG formalism and showed their effects in sentiment analysis.

Guo et al. applied their recursive neural networks to utterance classification and slot filling for spoken dialogue systems [16]. The recursive neural network for joint estimation of an utterance class and slot filling adopted two types of syntactic untying. One was syntactic type dependent tying and the other was dependent on syntactic types of the current and child nodes. The proposed model showed competitive performances with ATIS and Cortana data sets.

3. Data for Utterance Intent Classification

3.1 Air Travel Information System (ATIS)

Though the proposed method is targeted to process Japanese language, the general effectiveness in other languages is tested with the Air Travel Information System (ATIS) English data set. The ATIS has been widely used in SLU studies. ATIS data set has 18 intent classes: flight, airfare, ground service, airline and so on. The training set has 4,478 utterances from the ATIS-2 and ATIS-3 corpora, and the test set has 899 utterances from the ATIS-3 Nov93 and Dec04 data sets. Table 1 lists the intent classes with the relative frequency distribution and sample utterances. The relative frequency distribution has a greater imbalance than that of the Japanese data set, where the Flight class of the greatest frequencies occupies about three quarters of all the utterances. Meanwhile, the number of common words among different intent classes is greater than that in the Japanese data set.

Table 1 Utterance intent classes and relative frequencies of ATIS data set

Intent class tag	Freq.	Sample utterance
Flight	74.3	Flights from Newark to Boston.
Airfare	8.6	Show me the most expensive fare.
GroundService	5.2	List ground transportation in Detroit.
Airline	3.1	Which airlines serve Pittsburgh?
Abbreviation	2.9	What is air code H?
Aircraft	1.6	What planes are used by TWA?
FlightTime	1.0	What time does flight AA459 depart?
Quantity	0.9	How many booking classes are there?
City	0.4	What time zone is Denver in?
Distance	0.4	How far is Oakland airport from downtown?
Airport	0.4	What airport is at Tampa?
GroundFare	0.3	What are the rental car rates in Dallas?
Capacity	0.3	How many seats in a 734?
FlightNo	0.3	Flight numbers from Columbus to Minneapolis tomorrow.
Meal	0.1	What types of meals are available?
Restriction	0.1	What is restriction AP57?
DayName	0.1	What day of the week do flights from Nashville to Tacoma fly on?
Cheapest	0.1	Show me the cheapest fare in the database.

Freq.: relative frequency distribution in percent.

3.2 Smartphone-Based Japanese Spoken Dialogue System

The target system is a smartphone-based Japanese-language spoken dialogue application that was designed to encourage users to constantly use its speech interface [17]. The application introduced gamification to enhance the users' motivation to use the interface. In the beginning, the variety of responses from an animated character are severely limited, and the variation of responses and functionalities are gradually released with the continued use of the application. Major functionalities include weather forecasting, schedule managing, alarm setting, web searching, chatting, and so on.

Most of the user utterances are short phrases and words with a few sentences of complex ideas and nuances. The authors reviewed ASR log data of about 139,000 utterances, redefined utterance intent classes, and assigned one of the class tags to every utterance of part of the data. Specifically, three of the authors annotated the most frequent 3,000 variations of the ASR log data, which correspond to 97,000 utterances, i.e. 70.0% of the total. We redefined 169 utterance intent classes including an *others* class through discussions, and assigned a class tag to each of the 3,000 variations of utterances.

Frequent utterance intent classes out of the total of 169 classes, their relative frequency distribution and their sample utterances are listed in Table 2. Note that short sentences are selected as the sample utterances in the table due to space limitations. A small number of major classes have more than half of the total number of utterances, while a large number of minor classes have a small number of utterances.

Table 2 Utterance intent classes and their relative frequencies of smartphone-based Japanese spoken dialogue system

Intent class tag	Freq.	Sample utterance (translation)
CheckWeather	20.4	How's the weather in Tokyo now?
Greetings	16.5	Good morning.
AskTime	11.3	What time is it now?
CheckSchedule	7.2	Check today's schedule.
SetAlarm	5.7	Wake me up at 6am tomorrow.
Thanks	3.6	Thank you.
Yes	3.1	Yes.
Goodbye	2.4	Good night.
WebSearch	2.2	Search (keyword)
Praise	2.2	You are so cute.
Time	1.9	Tomorrow.
MakeFun	1.6	Stupid.
GoodFeeling	0.9	I'm fine.
BadFeeling	0.8	I am tired
CheckTemp	0.8	What is the temperature today?
BackChannel	0.7	Sure.
AddSchedule	0.7	Schedule a party at 7pm on Friday.
FortuneTeller	0.7	Tell my fortune today.
Call	0.6	Ho.
No	0.6	No way.

Freq.: relative frequency distribution in percent.

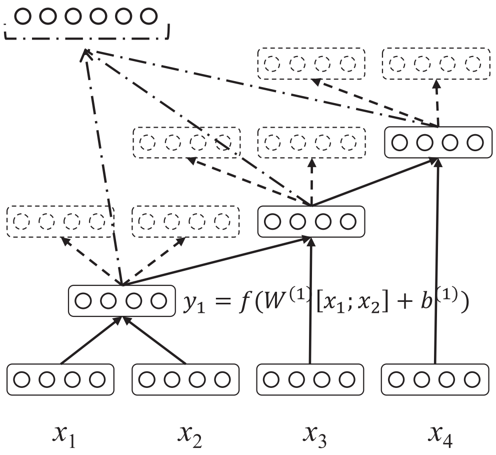

Fig. 1 Utterance intent classification with RAE. Solid lines represent RAE estimating continuous vector for non-leaf nodes and dashed-dotted lines represent softmax layer estimating intent class.

4. Intent Class Estimation Based on Untied RAE

4.1 Training of Basic RAE

Before proposing untying of the RAE, we start this section with an explanation of the basic RAE. Figure 1 illustrates the overall picture of RAE-based intent classification.

The classification based on RAE takes word embeddings as leaf nodes of a tree and applies an autoencoder to neighboring node pairs in a bottom-up manner repeatedly to form a tree. The RAE computes vectors of phrases and clauses at non-leaf nodes, and that of a whole utterance at the top node of the tree, The classification is performed by another softmax layer that takes all the vectors of the words, phrases, clauses and whole utterance as inputs and outputs a vector whose dimension is equal to the number of intent

Table 3 Manually designed node type table and autoencoder type table for Japanese.

Node type table

Node type index of left child node	Node type index of right child node												
	1	2	3	4	5	6	7	8	9	10	11	12	13
1	11	13	3	4	11	11	13	11	11	11	11	13	13
2	11	2	3	4	2	2	2	8	2	2	11	12	13
3	11	2	3	4	3	3	3	8	3	3	11	12	13
4	1	2	3	4	4	4	4	8	4	4	11	12	13
5	1	2	3	4	5	6	7	8	9	10	11	12	13
6	1	2	3	4	5	6	6	8	6	6	11	12	13
7	1	2	3	4	5	6	7	8	9	7	11	12	13
8	11	12	3	4	8	8	8	8	8	8	11	12	13
9	1	2	3	4	9	9	9	8	9	9	11	12	13
10	1	2	3	4	5	6	7	8	9	10	11	12	13
11	11	13	11	11	11	11	11	11	11	11	11	13	13
12	11	12	12	12	12	12	12	12	12	12	12	12	13
13	13	13	13	13	13	13	13	13	13	13	13	13	13
legend	1: noun, 2: verb, 3: adjective, 4: adverb, 5: particle, 6: conjunction, 7: auxiliary verb, 8: adnominal adjective, 9: interjection, 10: others, 11: noun phrase, 12: predicate phrase, 13: clause												

Autoencoder type table

Node type index	Autoencoder type index
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	1
9	1
10	1
11	1
12	2
13	2

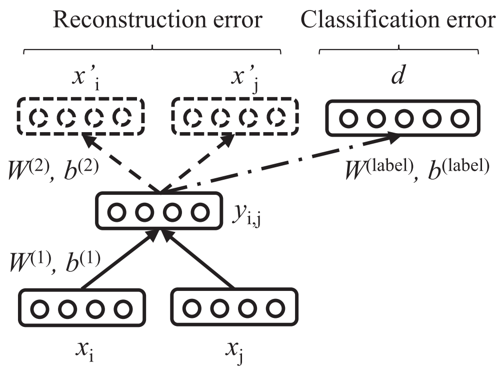

Fig. 2 Model parameters and error functions of RAE. Error functiions of RAE and softmax layer are reconstruction error and classification error, respectively.

classes.

The autoencoder applies a weighting matrix $W^{(1)}$ and bias $b^{(1)}$ with an activation function f to a vector pair of neighboring child nodes x_i and x_j, and outputs a composition vector $y_{(i,j)}$ with the same dimension as the parent node.

$$y_{(i,j)} = f(W^{(1)}[x_i; x_j] + b^{(1)}) \tag{1}$$

We use a sigmoid function as the activation function.

The autoencoder applies another neural network for an inversion which reproduces x_i and x_j as x'_i and x'_j from $y_{(i,j)}$ as accurately as possible. The inversion is expressed as Eq. (2).

$$[x'_i; x'_j] = \tilde{f}(W^{(2)}y_{(i,j)} + b^{(2)}) \tag{2}$$

We use the sigmoid function as the activation function $\tilde{f}$ for the inversion. The error function is reconstruction error E_{rec} expressed in Eq. (3).

$$E_{rec} = \frac{1}{2}|[x'_i; x'_j] - [x_i; x_j]|^2 \tag{3}$$

The tree is conceptually formed in accordance with a syntactic parse tree, but it is formed by a greedy search that minimizes the reconstruction error in reality. Among all pairs of neighboring nodes at a certain time, a pair with the minimal reconstruction error E_{rec} is selected to form a parent node.

Here, the autoencoder applied to every pair of nodes is a single common one; specifically, it is a set of model parameters $W^{(1)}$, $b^{(1)}$, $W^{(2)}$ and $b^{(2)}$. The set of model parameters of the tied RAE is trained to minimize the total of E_{rec} for all the training data.

The softmax layer for intent classification takes all the vectors of nodes as inputs, and outputs posterior probabilities of K units corresponding to the intent classes. The k'th component d_k of the output vector is expressed in Eq. (4).

$$d_k = \frac{exp(W_k^{(label)}y + b_k^{(label)})}{\sum_{j=1}^{K} exp(W_j^{(label)}y + b_j^{(label)})} \tag{4}$$

The correct signal is one hot vector.

$$t = [0, \ldots, 0, 1, 0, \ldots, 0]^T \tag{5}$$

The error function is the cross-entropy error E_{ce} expressed in Eq. (6).

$$E_{ce}(y, t) = -\sum_{k=1}^{K} t_k \log d_k(y) \tag{6}$$

Figure 2 lists the model parameters and error functions of the RAE. While the autoencoder aims to obtain a condensed vector representation best reproducing two child nodes of neighboring words or phrases, the whole RAE aims to classify the utterance intent accurately. As a whole, the total error function is set as a weighted sum of two error functions in Eq. (7).

$$E = \alpha E_{rec} + (1 - \alpha)E_{ce} \tag{7}$$

We set the weighting coefficient α to 0.2, the default value in [11] after confirming that it was reasonable in preliminary experiments.

The training of RAE optimizes the model parameters in accordance with the criterion of minimizing the total error function for all the training data.

4.2 Rule-Based Syntactic Untying of RAE

The basic RAE in the previous section applies a single arithmetic operation of an autoencoder to every pair of neighboring nodes. Even if the single autoencoder is optimized to reproduce any pair of child nodes well, the arithmetic operation is intrinsically different depending on the types of child nodes.

To reduce the difference of the nonlinear operation depending on the types of nodes, we manually designed a rule that switches two autoencoders depending on the types of two child nodes. We designed the rule for Japanese, not for English. At the leaf level of a tree, about a half of the words are nouns, while a sentence or phrase is composed of a predicate with a subject and/or objects and/or complements. The arithmetic operation of vectors between words and noun phrases, and that between predicate phrases and clauses are assumed to differ considerably. Hence, the manual rule switches two autoencoders; one for words and noun phrases and the other for predicate phrases and clauses. Along a tree, the former is applied at lower nodes around leaves, and the latter is applied at upper nodes close to the root node.

The node type is determined as follows. At leaf nodes, every word of a sentence is given a part-of-speech tag as a node type. Japanese sentences are processed by a Japanese morpheme analyzer [18]. The tag set for Japanese is comprised of ten part-of-speech tags: noun, verb, adjective, adverb, particle, auxiliary verb, adnominal adjective, conjunction, interjection and others. At upper nodes, a node type is determined by the combination of node types of two child nodes. Table 3 consists of two panels. The left panel shows the table determining the node type of a parent node based on the combination of the node types of left and right child nodes. This panel was defined by reference to a Japanese grammar textbook [19]. The right panel shows which autoencoder to apply based on the node type given in the left panel.

4.3 Data-Driven Untying of RAE

To obtain a more effective untied RAE, we designed a training method including data-driven untying of RAE. This method is based on splitting an autoencoder with a regression tree reducing the total reconstruction error E_{rec}. To be exact, this method alternates splitting an autoencoder into two with a binary regression tree with a response of the reconstruction error E_{rec} and optimizing the model parameters of the split autoencoders.

Figure 3 shows the procedure. The procedure starts with part-of-speech tagging of every morpheme of a sentence at preparation step 1). This part-of-speech tagging is the same process as that in the previous section for Japanese.

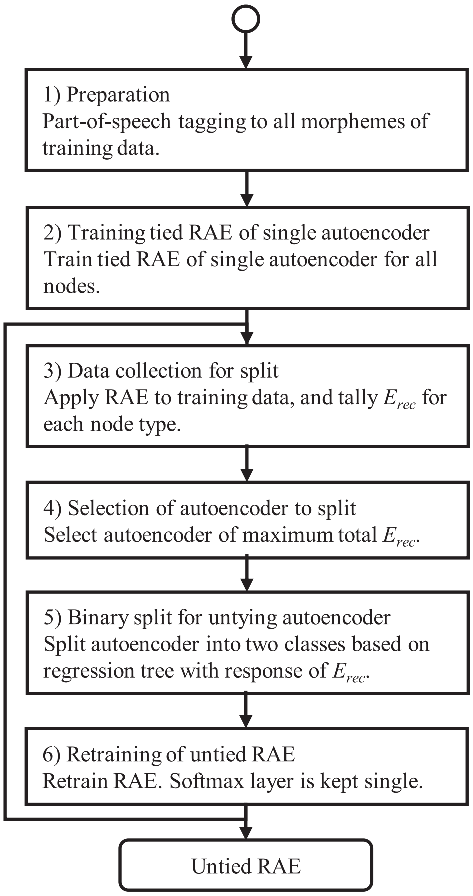
1) Preparation
Part-of-speech tagging to all morphemes of training data.
2) Training tied RAE of single autoencoder
Train tied RAE of single autoencoder for all nodes.
3) Data collection for split
Apply RAE to training data, and tally E_{rec} for each node type.
4) Selection of autoencoder to split
Select autoencoder of maximum total E_{rec}.
5) Binary split for untying autoencoder
Split autoencoder into two classes based on regression tree with response of E_{rec}.
6) Retraining of untied RAE
Retrain RAE. Softmax layer is kept single.
Untied RAE

Fig. 3 Procedure of training RAE of multiple autoencoders with data-driven untying.

For English, the sentences are first processed by a part-of-speech tagger in NLTK [20]. The tag set used is the 'universal' tag set with eleven tags but without the punctuation tag. The baseline bi-gram tagger was trained with 57,340 tagged sentences of the Brown corpus, and the Brill tagger was trained with 3,914 tagged sentences of the Treebank corpus and 577 tagged sentences of the ATIS corpus. Then, an initial tied RAE comprised of a single autoencoder is trained by the conventional method at step 2). In this training step, a tree is formed in the bottom-up manner for each sentence in the training data. While forming a tree, a node type is given to every node according to the node types of the child nodes. This is to be described in the next paragraph. The trained RAE is applied to the sentences of the training data, and the total reconstruction error E_{rec} is tallied for each autoencoder type; that is single in the initial RAE at step 3). Then, an autoencoder type of the maximum total reconstruction error E_{rec} is selected for splitting at step 4). At step 5), a class of all the node types pertaining to the selected autoencoder type is split into two based on a regression tree trained by CART [21] with a response of E_{rec}. The predictor variables are the node types of the left and right child nodes. The model parameters of the split autoencoders are initialized by those of the autoencoder before splitting and retrained with L2 regularization at step 6). After retraining,

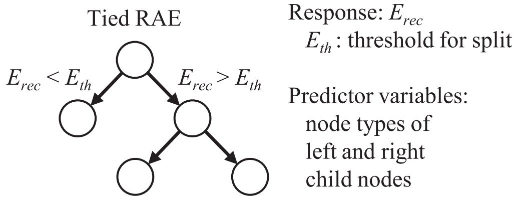

Fig. 4 Sequential splitting of autoencoders with response of E_{rec} and two predictor variables of node types of left and right child nodes.

Table 4 Node type index assignment table.

Node type of left child node		Node type of right child node							
		PoS index of leaf nodes					Autoencoder type of non-leaf nodes		
		1	2	·	·	10	11	12	13
PoS index of leaf nodes	1	-	-	-	-	-	-	-	-
	2	-	-	-	-	-	-	-	-
	·	-	-	Unique node type index of non-leaf nodes				-	-
	·	-	-					-	-
	10	-	-					-	-
Autoencoder type of non-leaf nodes	11	-	-					-	-
	12	-	-	-	-	-	-	-	-
	13	-	-	-	-	-	-	-	-

the RAE is applied to the sentences of the training data, and E_{rec} is tallied for each autoencoder type at step 3) again for the next splitting. The total reconstruction error E_{rec} can be used as a stopping condition of the iteration. In practice, the trained RAE was evaluated after each iteration this time. Figure 4 shows a regression tree produced by the sequential splitting. Each leaf of the tree corresponds to an autoencoder type. Here in the figure, the left node represents an autoencoder type of the smaller E_{rec}, while the right node represents an autoencoder type of the greater E_{rec} for the sake of simplicity. While the autoencoders are untied in a step-by-step manner, the softmax layer is kept single in order to avoid making the generated vector space completely different.

The data-driven assignment of the node types and autoencoder types has to be addressed in detail. At step 3), a node type index is given to every parent node in an automated way by referring to the node types of its child nodes. Table 4 illustrates how the node type index is given. For the leaf nodes, the part-of-speech tag index is used as the node type index. For the non-leaf nodes, a unique node type index is given to every combination of the node types of two child nodes appearing in the data incrementally. The rows and columns of the table represent the node types of the left child node and right child node, respectively. Any non-leaf node except for the root node becomes a child node of an upper node. The node type index of the non-leaf node, as either a left or right child node, is a unique index corresponding to its autoencoder type. The autoencoder type is determined by the data-driven splitting at step 5) in Fig. 3. The reason the autoencoder type is used as the node type of a child node instead of the node type index is to prevent an explosive increase of node type indices in the node type index assignment table in Table 4.

In the training of the initial tied RAE, the node types of the left and right child nodes are the part-of-speech indices of leaf nodes and one additional node type for all non-leaf nodes. The autoencoder type index assignment table is filled with a single autoencoder type index initially. Thereafter, the table is updated with the results of the data-driven splitting. This table is used for tallying the reconstruction error E_{rec} for each node type index, and the autoencoder type with the maximum total reconstruction error E_{rec} is chosen for the next split.

5. Experiments

5.1 Experimental Setup

We implemented the two untying methods by extending Socher's matlab implementation [11], and examined them with two data sets: the ATIS English data set and the smartphone-based Japanese spoken dialogue system data set.

Regarding the ATIS data set, we used the training set and test set as they were provided. The number of utterance intent classes was predefined as 18. The number of utterances in the training and test sets were 4,478 and 899, respectively.

Regarding the Japanese data set, the number of classes was reduced to 65 by merging classes with few pieces of data into a similar class or into the *other* class. By considering the balance of a few high-frequency utterances such as "What time is it now?" and a great number of low-frequency utterances, the frequencies of utterances were smoothed by taking their square root, and then placing the smoothed data set into the training set and test set randomly. The number of utterances in the training and test sets were 7,833 and 870, respectively. The fraction of unknown utterances in the test set was 15 percent.

5.2 Conditions of Experiments

We compared four methods with the English and Japanese data sets. The methods are 1) cosine similarity of bag-of-words (BoW), as a baseline, 2) tied RAE of a single autoencoder, 3) untied RAE based on manually-defined rule, and 4) untied RAE based on data-driven sequential split, which are the proposed methods. In the evaluation of the tied RAE, two types of word vectors, random vectors (2a) and word2vec vectors (2b), trained in the skip-gram mode were compared as the minimal components of a tree. The English word2vec vectors were trained with English Wikipedia texts of 3.5 billion words and had 3.25 million word entries. The Japanese word2vec vectors were trained with Japanese Wikipedia texts of 1.1 billion words and had 1.08 million word entries as a result. The dimension of the vectors was fixed at 100. Accordingly, the dimension of all the nodes was fixed at 100. The skip-gram mode for training word2vec

Table 5 Precision, recall, and accuracy of utterance intent classification of 18 classes of ATIS data set.

Method	Training set			Test set		
	prec.	recall	**acc.**	prec.	recall	**acc.**
1) Cosine similarity of bag-of-words (BoW)	-	-	-	59.8%	73.6%	**88.3%**
2a) Tied RAE based on random word vectors	60.9%	52.2%	**96.6%**	32.7%	32.8%	**87.2%**
2b) Tied RAE based on word2vec vectors	86.2%	77.6%	**97.4%**	54.5%	45.9%	**85.4%**
3) RAE of two autoencoders untied by manual rule	-	-	**-**	-	-	**-**
4a) RAE of two autoencoders untied by data-driven split	69.8%	67.3%	**97.0%**	54.7%	51.0%	**89.4%**
4b) RAE of three autoencoders untied by data-driven split	57.4%	58.7%	**96.5%**	44.7%	44.0%	**87.7%**

vectors was chosen based on the results of preliminary experiments.

Three types of RAEs, that is, a single tied autoencoder, two autoencoders untied by the manual rule, and multiple autoencoders untied by the data-driven sequential split were compared with the baseline method of cosine similarity of bag-of-word vectors.

5.3 Experimental Results

Table 5 shows the precision, recall, and accuracy of the classification for the training and test sets of the ATIS English data set. The baseline BoW method with maximization of cosine similarity (1) showed a relatively high performance. We consider the reason is the test set contained a great ratio of known words. The tied RAE based on random word vectors (2a) showed a higher accuracy (87.2%) than the tied RAE based on word2vec vectors (2b) for the test set, but method (2b) showed higher performance than method (2a) in precision and recall values. We did not test method (3) for the ATIS data set because it was difficult to define a manual rule for English. The RAE of two autoencoders untied by data-driven split (4a) showed the best accuracy, and the RAE of three autoencoders untied by data-driven split (4b) showed a decrease. We consider that the RAE was overlearned with thousands pieces of training data. We conducted a sign test for significant difference between the methods. A significant difference was not observed between method (1) and (4a) with a significant level of 0.1, but a significant difference was observed between method (2b) and (4a) with a significant level of 0.01.

Figure 5 shows the regression tree generated by the data-driven split for the ATIS data set. In common with Fig. 4, the left and right nodes in every split represent the autoencoder types of the smaller and the greater E_{rec}, respectively. The first split was made on the type of the left child node. Non-leaf nodes with an adposition or a particle or a determinant or a pronoun as its left child node had a smaller reconstruction error E_{rec}, while non-leaf nodes with a noun or a verb or an adjective or an adverb or a conjunction or a numeral or an other or a non-leaf node as its left child node had a greater reconstruction error E_{rec}. This split is understood as a separation of adding a determiner from the others.

Table 6 shows the precision, recall, and accuracy of the classification for the training and test sets of the Japanese data set. The baseline method (1) showed a relatively high performance. We consider the reason is the test set randomly chosen considering the smoothed frequencies contained a great ratio of known words and utterances that were seen in the training set. The tied RAE based on word2vec vectors (2b) showed significantly better performance than the tied RAE based on random word vectors (2a). While the RAE of two autoencoders untied by a manual rule (3) made a slight improvement in performance, the RAE of two autoencoders untied by data-driven split (4a) made a larger improvement. However, the RAE of three autoencoders untied by data-driven split (4b) showed a fall as in the case of the ATIS data set. We conducted a sign test for significant difference between the methods. A significant difference was not observed between method (1) and (4a) with a significant level of 0.1, but significant differences were observed between method (2b) and (4a) with a significant level of 0.01 and between method (3) and (4a) with a significant level of 0.05, respectively.

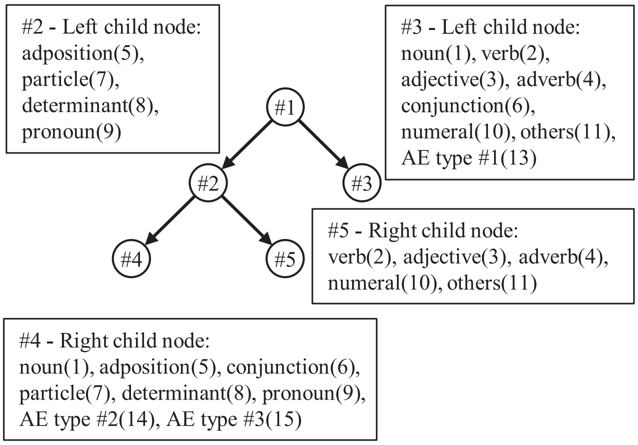

Fig. 5 Generated regression tree for untying RAE with ATIS data set.

Figure 6 shows the regression tree generated by the data-driven split. The first split was made on the type of the right child node. Non-leaf nodes with a noun or an adjective or an adverb or a particle or an interjection or a non-leaf node as its right child node had a smaller reconstruction error E_{rec}, while non-leaf nodes with a verb or a conjunction or an auxiliary verb or an adnominal adjective or an other as its right child node had a greater reconstruction error E_{rec}. It is not easy to characterize this split simply, but this split looks like a separation of predicates from the others.

Figure 7 shows how the total reconstruction error E_{rec} decreases with respect to the number of autoencoder types in

Table 6 Precision, recall, and accuracy of utterance intent classification of 65 classes of Japanese data set.

Method	Training set			Test set		
	prec.	recall	**acc.**	prec.	recall	**acc.**
1) Cosine similarity of bag-of-words (BoW)	-	-	-	76.0%	74.2%	**85.1%**
2a) Tied RAE based on random word vectors	37.2%	33.2%	**70.6%**	32.0%	65.6%	**66.4%**
2b) Tied RAE based on word2vec vectors	81.2%	78.8%	**88.7%**	74.7%	70.5%	**82.7%**
3) RAE of two autoencoders untied by manual rule	65.9%	68.3%	**88.1%**	63.0%	62.5%	**84.0%**
4a) RAE of two autoencoders untied by data-driven split	80.3%	79.8%	**91.3%**	72.4%	72.3%	**85.6%**
4b) RAE of three autoencoders untied by data-driven split	73.9%	75.2%	**90.3%**	70.8%	67.9%	**84.8%**

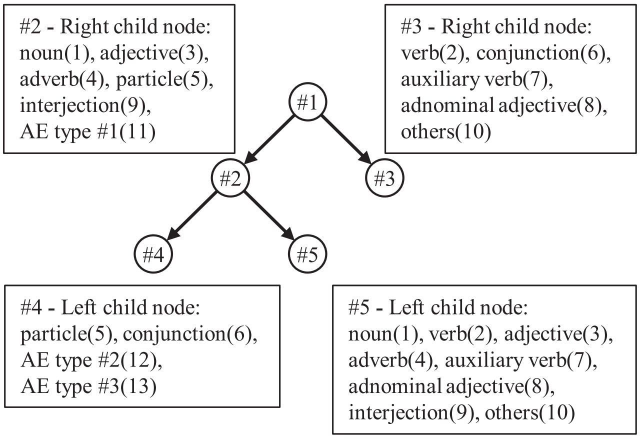

Fig. 6 Generated regression tree for untying RAE with Japanese data set.

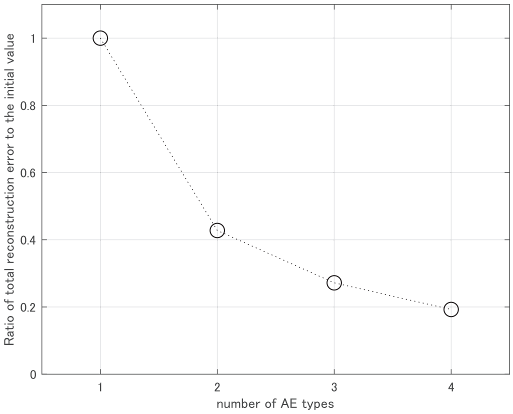

Fig. 7 Decrease of total reconstruction error E_{rec} w.r.t the number of autoencoder types in data-driven untying of RAE.

the data-driven splitting process. The horizontal and vertical axes represent the number of autoencoder types and the ratio of total reconstruction error to its initial value, respectively. The total reconstruction error decreased the most in the first split, and little by little after the second split as is expected.

6. Conclusions

To provide flexibility and efficiency to the RAE model for SLU for spoken dialogue systems, an efficient data-driven untying of the RAE is proposed and examined with two utterance intent classification tasks of English and Japanese spoken dialogue systems. It is difficult to design an untying of RAE manually in practice, but the data-driven split with a criterion of minimizing the reconstruction error using the PoS information improved the accuracy. The regression trees generated by this method showed reasonable splits, that is, separating addition of a determiner at the left child node first for English and separating predicates from the others at the right child node first for Japanese.

References

[1] V.N. Vapnik, Statistical Learning Theory, John Wiley and Sons, New York, NY, 1998.

[2] P. Haffner, G. Tur, and J.H. Wright, “Optimizing SVMs for Complex Call Classification,” Proc. of ICASSP 2003, vol.1, pp.632–635, 2003.

[3] C. Chelba, M. Mahajan, and A. Acero, “Speech Utterance Classification,” Proc. of ICASSP 2003, vol.1, pp.280–283, 2003.

[4] G.A. Miller, “WordNet: A Lexical Database for English,” Communication of the ACM, vol.38, no.11, pp.39–41, 1995.

[5] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, “Efficient estimation of word representation in vector space,” arXiv: 1301.3781, 2013.

[6] J. Pennington, R. Socher, and C.D. Manning, “GloVe: Global Vectors for Word Representation,” Proc. of EMNLP 2014, pp.1532–1543, 2014.

[7] Y. Luan, S. Watanabe, and B. Harsham, “Efficient learning for spoken language understanding tasks with word embedding based pretraining,” Proc. Interspeech 2015, pp.1398–1402, 2015.

[8] B. Liu and I. Lane, “Joint Online Spoken Language Understanding and Language Modeling with Recurrent Neural Networks,” Proc. of SIGDIAL 2016, pp.22–30, 2016.

[9] B. Liu and I. Lane, “Attention-Based Recurrent Neural Network Models for Joint Intent Detection and Slot Filling,” Proc. of Interspeech 2016, pp.685–689, 2016.

[10] Y.-N. Chen, D. Hakanni-Tür, G. Tur, A. Celikyilmaz, J. Guo, and L. Deng, “Syntax or Semantics? Knowledge-guided Joint Semantic Frame Parsing,” Proc. of Spoken Language Technology Workshop 2016, pp.348–354, 2016.

[11] R. Socher, J. Pennington, E.H. Huang, A.Y. Ng, and C.D. Manning, “Semi-supervised recursive autoencoders for predicting sentiment distributions,” Proc. EMNLP 2011, pp.151–161, 2011.

[12] R. Socher, B. Huval, C.D. Manning, and A.Y. Ng, “Semantic compositionality through recursive matrix vector spaces,” Proc. EMNLP 2012, pp.1631–1642, 2012.

[13] R. Socher, J. Bauer, C.D. Manning, and A.Y. Ng, “Parsing with compositional vector grammars,” Proc. ACL 2013, pp.455–465, 2013.

[14] A. Szabolcsi, “Bound variables in syntax: Are they any?,” Semantics and Contextual Expression, pp.295–318, 1989.

[15] K.M. Hermann and P. Blunsom, “The role of syntax in vector space models of compositional semantics,” Proc. ACL 2013, pp.894–904, 2013.

[16] D. Guo, G. Tur, W. Yih, and G. Zweig, “Joint Semantic Utterance Classification and Slot Filling with Recursive Neural Networks,” Proc. of Spoken Language Technology Workshop 2014,

pp.554–559, 2014.

[17] X. Xu, J. Wu, K. Fujita, T. Kato, and F. Sugaya, “Hey Peratama: a Breeding Game with Spoken Dialogue Interface,” Proc. of International Conference on Mobile and Ubiquitous Multimedia 2014, pp.266–267, 2014.

[18] T. Kudo, K. Yamamoto, and Y. Matsumoto, “Applying conditional random fields to Japanese morphological analysis,” Proc. EMNLP 2004, pp.230–237, 2004.

[19] T. Masuoka and Y. Takubo, Kiso Nihongo Bunpo (“Fundamental Japanese Grammar” in Japanese), Kuroshio Shuppan, 1992.

[20] S. Bird, E. Klein, and E. Loper, Natural Language Processing with Python, O’Reilly Media, Inc., 2009.

[21] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone, Classification and Regression Trees, Chapman & Hall CRC, 1984.

Tsuneo Kato received his B.E., M.E., and Ph.D. degrees from The University of Tokyo in 1994, 1996, and 2011. He joined Doshisha University in 2015, where he is currently an associate professor in the Department of Intelligent Information Engineering and Sciences, Faculty of Science and Engineering. Before his current position, he had worked at KDDI R&D Laboratories Inc. since 1996. He has been engaged in research and development of automatic speech recognition and intelligent user interfaces. He received an IPSJ Kiyasu Special Industrial Achievement Award in 2011. He is a member of IEICE, ANLP, ASJ, IPSJ, ACM and IEEE.

Atsushi Nagai received his B.S. degree from Doshisha University in 2017. He is currently with the Graduate School of Science and Engineering, Doshisha University. He is a member of ANLP.

Naoki Noda received his B.S. degree from Doshisha University in 2017. He is currently with the Graduate School of Science and Engineering, Doshisha University. He is a member of IEICE.

Jianming Wu received his B.E. degree from Shanghai Jiaotong University in 1998. He received his M.E. and Ph.D. degrees from Waseda University in 2002, 2005 respectively. He joined KDDI R&D Laboratories Inc. since 2005, where he is currently an R&D Manager in the Department of Intelligent Media Laboratory of KDDI Research Inc. He has been engaged in mobile OS, multi-device dialogue system, facial expression recognition and chatbot AI. He is a member of IEICE and IPSJ.

Seiichi Yamamoto received his B.S., M.S., and Ph.D. degrees from Osaka University in 1972, 1974, and 1983. He joined Kokusai Denshin Denwa Co. Ltd. in April 1974 and ATR Interpreting Telecommunications Research Laboratories in May 1997. He was appointed president of ATR-ITL in 1997. He is currently a professor in the Department of Information Systems Design, Faculty of Science and Engineering, Doshisha University, Kyoto, Japan. His research interests include digital signal processing, speech recognition, speech synthesis, natural language processing, spoken language processing, spoken language translation, and multi-modal dialogue processing. He received Technology Development Awards from the Acoustical Society of Japan in 1995 and 1997, a best paper award from the Information and Systems Society of IEICE in 2006, and a telecom-system technology award from the Telecommunications Advancement Foundation in 2007. Dr. Yamamoto is a member of ASJ, IPSJ, IEEE (Fellow), and IEICE Japan (Fellow).