Retrieval Term Prediction Using Deep Belief Networks

Qing Ma[†] **Ibuki Tanigawa**[†] **Masaki Murata**[‡]
[†] Department of Applied Mathematics and Informatics, Ryukoku University
[‡] Department of Information and Electronics, Tottori University
qma@math.ryukoku.ac.jp

Abstract

This paper presents a method to predict retrieval terms from relevant/surrounding words or descriptive texts in Japanese by using deep belief networks (DBN), one of two typical types of deep learning. To determine the effectiveness of using DBN for this task, we tested it along with baseline methods using example-based approaches and conventional machine learning methods, i.e., multi-layer perceptron (MLP) and support vector machines (SVM), for comparison. The data for training and testing were obtained from the Web in manual and automatic manners. Automatically created pseudo data was also used. A grid search was adopted for obtaining the optimal hyper-parameters of these machine learning methods by performing cross-validation on training data. Experimental results showed that (1) using DBN has far higher prediction precisions than using baseline methods and higher prediction precisions than using either MLP or SVM; (2) adding automatically gathered data and pseudo data to the manually gathered data as training data is an effective measure for further improving the prediction precisions; and (3) DBN is able to deal with noisier training data than MLP, i.e., the prediction precision of DBN can be improved by adding noisy training data, but that of MLP cannot be.

1 Introduction

The current Web search engines have a very high retrieval performance as long as the proper retrieval terms are given. However, many people, particularly children, seniors, and foreigners, have difficulty deciding on the proper retrieval terms for representing the retrieval objects,[1] especially with searches related to technical fields. The support systems are in place for search engine users that show suitable retrieval term candidates when some clues such as their descriptive texts or relevant/surrounding words are given by the users. For example, when the relevant/surrounding words "computer", "previous state", and "return" are given by users, "system restore" is predicted by the systems as a retrieval term candidate.

Our objective is to develop various domain-specific information retrieval support systems that can predict suitable retrieval terms from relevant/surrounding words or descriptive texts in Japanese. To our knowledge, no such studies have been done so far in Japanese. As the first step, here, we confined the retrieval terms to the computer-related field and proposed a method to predict them using machine learning methods with deep belief networks (DBN), one of two typical types of deep learning.

In recent years, deep learning/neural network techniques have attracted a great deal of attention in various fields and have been successfully applied not only in speech recognition (Li et al., 2013) and image recognition (Krizhevsky et al., 2012) tasks but also in NLP tasks including morphology & syn-

[1] For example, according to a questionnaire administered by Microsoft in 2010, about 60% of users had difficulty deciding on the proper retrieval terms. (http://www.garbagenews.net/archives/1466626.html) (http://news.mynavi.jp/news/2010/07/05/028/)

28th Pacific Asia Conference on Language, Information and Computation pages 338–347

tax (Billingsley and Curran, 2012; Hermann and Blunsom, 2013; Luong et al., 2013; Socher et al., 2013a), semantics (Hashimoto et al., 2013; Srivastava et al., 2013; Tsubaki et al., 2013), machine translation (Auli et al., 2013; Liu et al., 2013; Kalchbrenner and Blunsom, 2013; Zou et al., 2013), text classification (Glorot et al., 2011), information retrieval (Huang et al., 2013; Salakhutdinov and Hinton, 2009), and others (Seide et al., 2011; Socher et al., 2011; Socher et al., 2013b). Moreover, a unified neural network architecture and learning algorithm has also been proposed that can be applied to various NLP tasks including part-of-speech tagging, chunking, named entity recognition, and semantic role labeling (Collobert et al., 2011).

To our knowledge, however, there have been no studies on applying deep learning to information retrieval support tasks. We therefore have two main objectives in our current study. One is to develop an effective method for predicting suitable retrieval terms and the other is to determine whether deep learning is more effective than other conventional machine learning methods, i.e., multi-layer perceptron (MLP) and support vector machines (SVM), in such NLP tasks.

The data used for experiments were obtained from the Web in both manual and automatic manners. Automatically created pseudo data was also used. A grid search was used to obtain the optimal hyperparameters of these machine learning methods by performing cross-validation on training data. Experimental results showed that (1) using DBN has a far higher prediction precision than using baseline methods and a higher prediction precision than using either MLP or SVM; (2) adding automatically gathered data and pseudo data to the manually gathered data as training data is an effective measure for further improving the prediction precision; and (3) the DBN can deal with noisier training data than the MLP, i.e., the prediction precision of DBN can be improved by adding noisy training data, but that of MLP cannot be.

2 The Corpus

For training, a corpus consisting of pairs of inputs and their responses (or correct answers) — in our case, pairs of the relevant/surrounding words or descriptive texts and retrieval terms — is needed. The responses are typically called labels in supervised learning and so here we call the retrieval terms labels. Table 1 shows examples of these pairs, where the "Relevant/surrounding words" are those extracted from descriptive texts in accordance with steps described in Subsection 2.4. In this section, we describe how the corpus is obtained and how the feature vectors of the inputs are constructed from the corpus for machine learning.

2.1 Manual and Automatic Gathering of Data

Considering that the descriptive texts of labels necessarily include their relevant/surrounding words, we gather Web pages containing these texts in both manual and automatic manners. In the manual manner, we manually select the Web pages that describe the labels. In contrast, in the automatic manner, we respectively combine five words or parts of phrases とは (*toha*, "what is"), は (*ha*, "is"), というものは (*toiumonoha*, "something like"), については (*nitsuiteha*, "about"), and の意味は (*noimiha*, "the meaning of"), on the labels to form the retrieval terms (e.g., if a label is グラフィックボード (*gurafikku boudo*, "graphic board"), then the retrieval terms are グラフィックボード とは (*gurafikku boudo toha*, "what is graphic board"), グラフィックボード は (*gurafikku boudo ha*, "graphic board is"), and etc.) and then use these terms to obtain the relevant Web pages by a Google search.

2.2 Pseudo Data

To acquire as high a generalization capability as possible, for training we use not only the small scale of manually gathered data, which is high precision, but also the large scale of automatically gathered data, which includes a certain amount of noise. In contrast to manually gathered data, automatically gathered data might have incorrect labels, i.e., labels that do not match the descriptive texts. We therefore also use pseudo data, which can be regarded as data that includes some noises and/or deficiencies added to the original data (i.e., to the descriptive texts of the manually gathered data) but with less noise than the automatically gathered data and with all the labels correct. The procedure for creating pseudo data from the manually gathered data involves (1) extracting all the different words from the

Labels (Retrieval terms)	Inputs (Descriptive texts or relevant/surrounding words; translated from Japanese)	
Graphic board	**Descriptive text**	Also known as: graphic card, graphic accelerator, GB, VGA. While the screen outputs the picture actually seen by the eye, the screen only displays as commanded and does not output anything if ···
	Relevant/surrounding words	screen, picture, eye, displays, as commanded, ···
	Descriptive text	A device that provides independent functions for outputting or inputting video as signals on a PC or various other types of computer ···
	Relevant/surrounding words	independent, functions, outputting, inputting, video, signals, PC, ···
Main memory	···	···

Table 1: Examples of input-label pairs in the corpus.

manually gathered data and (2) for each label, randomly adding the words that were extracted in step (1) but not included in the descriptive texts and/or deleting words that originally existed in the descriptive texts so that the newly generated data (i.e., the newly generated descriptive texts) have 10% noises and/or deficiencies added to the original data.

2.3 Testing Data

The data described in Subsections 2.1 and 2.2 are for training. The data used for testing are different to the training data and are also obtained from automatically gathered data. Since automatically gathered data may include a lot of incorrect labels that cannot be used as objective assessment data, we manually select correct ones from the automatically gathered data.

2.4 Word Extraction and Feature Vector Construction

Relevant/surrounding words are extracted from descriptive texts by steps (1)–(4) below and the inputs are represented by feature vectors in machine learning constructed by steps (1)–(6): (1) perform morphological analysis on the manually gathered data and extract all nouns, including proper nouns, verbal nouns (nouns forming verbs by adding word する (*suru*, "do")), and general nouns; (2) connect the nouns successively appearing as single words; (3) extract the words whose appearance frequency in each label is ranked in the top 50; (4) exclude the words appearing in the descriptive texts of more than two labels; (5) use the words obtained by the above steps as the vector elements with binary values, taking value 1 if a word appears and 0 if not; and (6) perform morphological analysis on all data described in Subsections 2.1, 2.2, and 2.3 and construct the feature vectors in accordance with step (5).

3 Deep Learning

Two typical approaches have been proposed for implementing deep learning: using deep belief networks (DBN) (Hinton et al., 2006; Lee et al., 2009; Bengio et al., 2007; Bengio, 2009; Bengio et al., 2013) and using stacked denoising autoencoder (SdA) (Bengio et al., 2007; Bengio, 2009; Bengio et al., 2013; Vincent et al., 2008; Vincent et al., 2010). In this work we use DBN, which has an elegant architecture and a performance more than or equal to that of SdA in many tasks.

DBN is a multiple layer neural network equipped with an unsupervised learning based on restricted Boltzmann machines (RBM) for pre-training to extract features and a supervised learning for fine-tuning to output labels. The supervised learning can be implemented with a single layer or multi-layer perceptron or others (linear regression, logistic regression, etc.).

3.1 Restricted Boltzmann Machine

RBM is a probabilistic graphical model representing the probability distribution of training data with a fast unsupervised learning.

It consists of two layers, one visible and

one hidden, that respectively have visible units $(v_1, v_2, \cdots, v_m)$ and hidden units $(h_1, h_2, \cdots, h_n)$ connected to each other between the two layers (Figure 1).

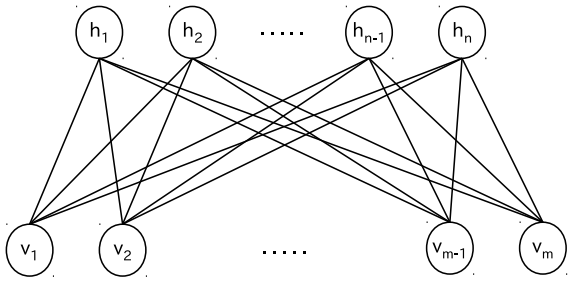


Figure 1: Restricted Boltzmann machine.

Given training data, the weights of the connections between units are modified by learning so that the behavior of the RBM stochastically fits the training data as well as possible. The learning algorithm is briefly described below.

First, sampling is performed on the basis of conditional probabilities when a piece of training data is given to the visible layer using Eqs. (1), (2), and then (1) again:

$$P(h_i^{(k)} = 1|\boldsymbol{v}^{(k)}) = \text{sigmoid}(\sum_{j=1}^{m} w_{ij} v_j^{(k)} + c_i) \quad (1)$$

and

$$P(v_j^{(k+1)} = 1|\boldsymbol{h}^{(k)}) = \text{sigmoid}(\sum_{i=1}^{n} w_{ij} h_i^{(k)} + b_j), \quad (2)$$

where k (≥ 1) is a repeat count of sampling and $\boldsymbol{v}^{(1)} = \boldsymbol{v}$ which is a piece of training data, w_{ij} is the weight of connection between units v_j and h_i, and b_j and c_i are offsets for the units v_j and h_i of the visible and hidden layers. After k repetition sampling, the weights and offsets are updated by

$$\boldsymbol{W} \leftarrow \boldsymbol{W} + \epsilon(\boldsymbol{h}^{(1)}\boldsymbol{v}^T - P(\boldsymbol{h}^{(k+1)} = 1|\boldsymbol{v}^{(k+1)})\boldsymbol{v}^{(k+1)T}), \quad (3)$$

$$\boldsymbol{b} \leftarrow \boldsymbol{b} + \epsilon(\boldsymbol{v} - \boldsymbol{v}^{(k+1)}), \quad (4)$$

$$\boldsymbol{c} \leftarrow \boldsymbol{c} + \epsilon(\boldsymbol{h}^{(1)} - P(\boldsymbol{h}^{(k+1)} = 1|\boldsymbol{v}^{(k+1)})), \quad (5)$$

where ϵ is a learning rate and the initial values of $\boldsymbol{W}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ are $\boldsymbol{0}$. Sampling with a large enough repeat count is called Gibbs sampling, which is computationally expensive. A method called k-step Contrastive Divergence (CD-k) which stops sampling after k repetitions is therefore usually adopted. It is empirically known that even $k = 1$ (CD-1) often gives good results, and so we set $k = 1$ in this work.

If we assume totally e epochs are performed for learning n training data using CD-k, the procedure for learning RBM can be given as in Figure 2. As the learning progresses, the samples[2] of the visible layer $\boldsymbol{v}^{(k+1)}$ approach the training data $\boldsymbol{v}$.

For each of all epochs e **do**
  For each of all data n **do**
    For each repetition of CD k **do**
      Sample according to Eqs. (1), (2), (1)
    End for
    Update using Eqs. (3), (4), (5)
  End for
End for

Figure 2: Procedure for learning RBM.

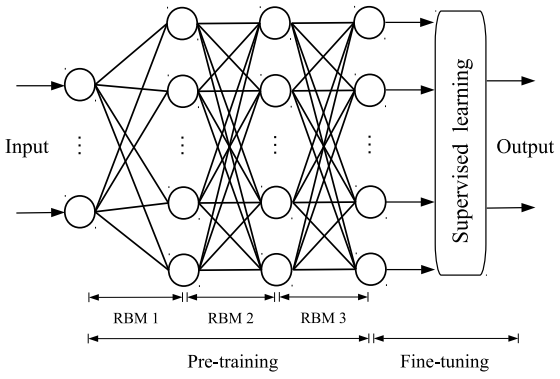


Figure 3: Example of a deep belief network.

3.2 Deep Belief Network

Figure 3 shows a DBN composed of three RBMs for pre-training and a supervised learning device for fine-tuning. Naturally the number of RBMs is changeable as needed. As shown in the figure, the hidden layers of the earlier RBMs become the visible layers of the new RBMs. Below, for simplic-

[2]By "samples" here we mean the data generated on the basis of the conditional probabilities of Eqs. (1) and (2).

ity, we consider the layers of RBMs (excluding the input layer) as hidden layers of DBN. The DBN in the figure therefore has three hidden layers, and this number is equal to the number of RBMs. Although supervised learning can be implemented by any method, in this work we use logistic regression.

The procedure for learning the DBN with three RBMs is shown in Figure 4.

1. Train RBM 1 with the training data as inputs by the **procedure for learning RBM (Figure 2)** and fix its weights and offsets.
2. Train RBM 2 with the samples of the hidden layer of RBM 1 as inputs by the **procedure for learning RBM (Figure 2)** and fix its weights and offsets.
3. Train RBM 3 with the samples of the hidden layer of RBM 2 as inputs by the **procedure for learning RBM (Figure 2)** and fix its weights and offsets.
4. Perform supervised learning with the samples of the hidden layer of RBM 3 as inputs and the labels as the desired outputs.

Figure 4: Procedure for learning DBN with three RBMs.

4 Experiments

4.1 Experimental Setup

4.1.1 Data

We formed 13 training data sets by adding different amounts of automatically gathered data and/or pseudo data to a base data set, as shown in Table 2. In the table, m300 is the base data set including 300 pieces of manually gathered data and, for example, a2400 is a data set including 2,400 automatically gathered pieces of data and m300, p2400 is a data set including 2,400 pieces of pseudo data and m300, and a2400p2400 is a data set including 2,400 pieces of automatically gathered data, 2,400 pieces of pseudo data, and m300. Altogether there were 100 pieces of testing data. The number of labels was 10; i.e., the training data listed in Table 2 and the testing data have 10 labels. The dimension of the feature vectors constructed in accordance with the steps in Subsection 2.4 was 182.

m300			
a300	a600	a1200	a2400
p300	p600	p1200	p2400
a300p300	a600p600	a1200p1200	a2400p2400

Table 2: Training data sets.

4.1.2 Hyperparameter Search

The optimal hyperparameters of the various machine learning methods used were determined by a grid search using 5-fold cross-validation on training data. The hyperparameters for the grid search are shown in Table 3. To avoid unfair bias toward the DBN during cross-validation due to the DBN having more hyperparameters than the other methods, we divided the MLP and SVM hyperparameter grids more finely than that of the DBN so that they had the same or more hyperparameter combinations than the DBN. For MLP, we also considered another case in which we used network structures, learning rates, and learning epochs completely the same as those of the DBN. In this case, the number of MLP hyperparameter combinations was quite small compared to that of the DBN. We refer to this MLP as MLP 1 and to the former MLP as MLP 2. Ultimately, the DBN and MLP 2 both had 864 hyperparameter combinations, the SVM (Linear) and SVM (RBF) had 900, and MLP 1 had 72.

4.1.3 Baselines

For comparison, in addition to MLP and SVM, we run tests on baseline methods using example-based approaches and compare the testing data of each with all the training data to determine which one had the largest number of words corresponding to the testing data. The algorithm is shown in Figure 5, where the words used for counting are those extracted from the descriptive texts in accordance with steps (1)–(4) in Subsection 2.4.

[3] As an example, the structure (hidden layers) **152-121-91** shown in the table refers to a DBN with a 182-**152-121-91**-10 structure, where 182 and 10 refer to dimensions of the input and output layers, respectively. These figures were set not in an arbitrary manner but using regular intervals in a linear form, i.e., $152 = 182 \times 5/6, 121 = 182 \times 4/6$, and $91 = 182 \times 3/6$.

Machine learning methods	Hyperparameters	Values
DBN	structure (hidden layers)[3]	91, 137-91, 152-121-91, 273, 273-273, 273-273-273
	ϵ of pre-training	0.001, 0.01, 0.1
	ϵ of fine-tuning	0.001, 0.01, 0.1
	epoch of pre-training	500, 1000, 2000, 3000
	epoch of fine-tuning	500, 1000, 2000, 3000
MLP 1	structure (hidden layers)	91, 137-91, 152-121-91, 273, 273-273, 273-273-273
	ϵ	0.001, 0.01, 0.1
	epoch	500, 1000, 2000, 3000
MLP 2	structure (hidden layers)	91, 137-91, 152-121-91, 273, 273-273, 273-273-273
	ϵ	0.001, 0.0025, 0.005, 0.0075, 0.01, 0.025, 0.05, 0.075, 0.1
	epoch	6 divisions between 500-1000 and 10 divisions between 1200-3000 in a linear scale
SVM (Linear)	γ	900 divisions between 10^{-4}-10^{4} in a logarithmic scale
SVM (RBF)	γ	30 divisions between 10^{-4}-10^{4} in a logarithmic scale
	C	30 divisions between 10^{-4}-10^{4} in a logarithmic scale

Table 3: Hyperparameters for grid search.

For each input **i** of testing data **do**
  For each input **j** of training data **do**
    1. Count the same words between **i** and **j**
    2. Find the **j** with the largest count and set **m=j**
  End for
  1. Let the label of **m** of training data (**r**) be the predicting result of the input **i**
  2. Compare **r** with the label of **i** of testing data and determine the correctness
End for
1. Count the correct predicting results and compute the correct rate (precision)

Figure 5: Baseline algorithm.

4.2 Results

Figure 6 compares the testing data precisions when using different training data sets with individual machine learning methods. The precisions are averages when using the top N sets of the hyperparameters in ascending order of the cross-validation errors, with N varying from 5 to 30.

As shown in the figure, both the DBN and the MLPs had the highest precisions overall and the SVMs had approximately the highest precision when using data set a2400p2400, i.e., in the case of adding the largest number of automatically gathered data and pseudo data to the manually gathered data as training data. Moreover, the DBN, MLPs, and SVM (RBF) all had higher precisions when adding the appropriate amount of automatically gathered data and pseudo data compared to the case of using only manually gathered data, but the SVM (Linear) did not have this tendency.[4] Further, the DBN and SVM (RBF) had higher precisions when adding the appropriate amount of automatically gathered data only, whereas the MLPs had higher precisions when adding the appropriate amount of pseudo data only compared to the case of using only manually gathered data. From these results, we can infer that (1) all the machine learning methods (excluding SVM (Linear)) can improve their precisions by adding automatically gathered and pseudo data as training data and that (2) the DBN and SVM (RBF) can deal with noisier data than the MLPs, as the automatically gathered data are noisier than the pseudo data.

Figure 7 compares the testing data precisions of DBN and MLPs and of DBN and SVMs when using different training data sets (i.e., the data set of Table 2) that are not distinguished from each other. As in Figure 6, the precisions are averages of using the top N sets of hyperparameters in ascending order of the cross-validation errors, with N varying from 5 to 30. We can see at a glance that the performance of the DBN was generally superior to all the other machine learning methods. We should point out that the ranges of the vertical axes of all the graphs are set to be the same and so four lines of the SVM (RBF)

[4]This is because the SVM (Linear) can only deal with data capable of linear separation.

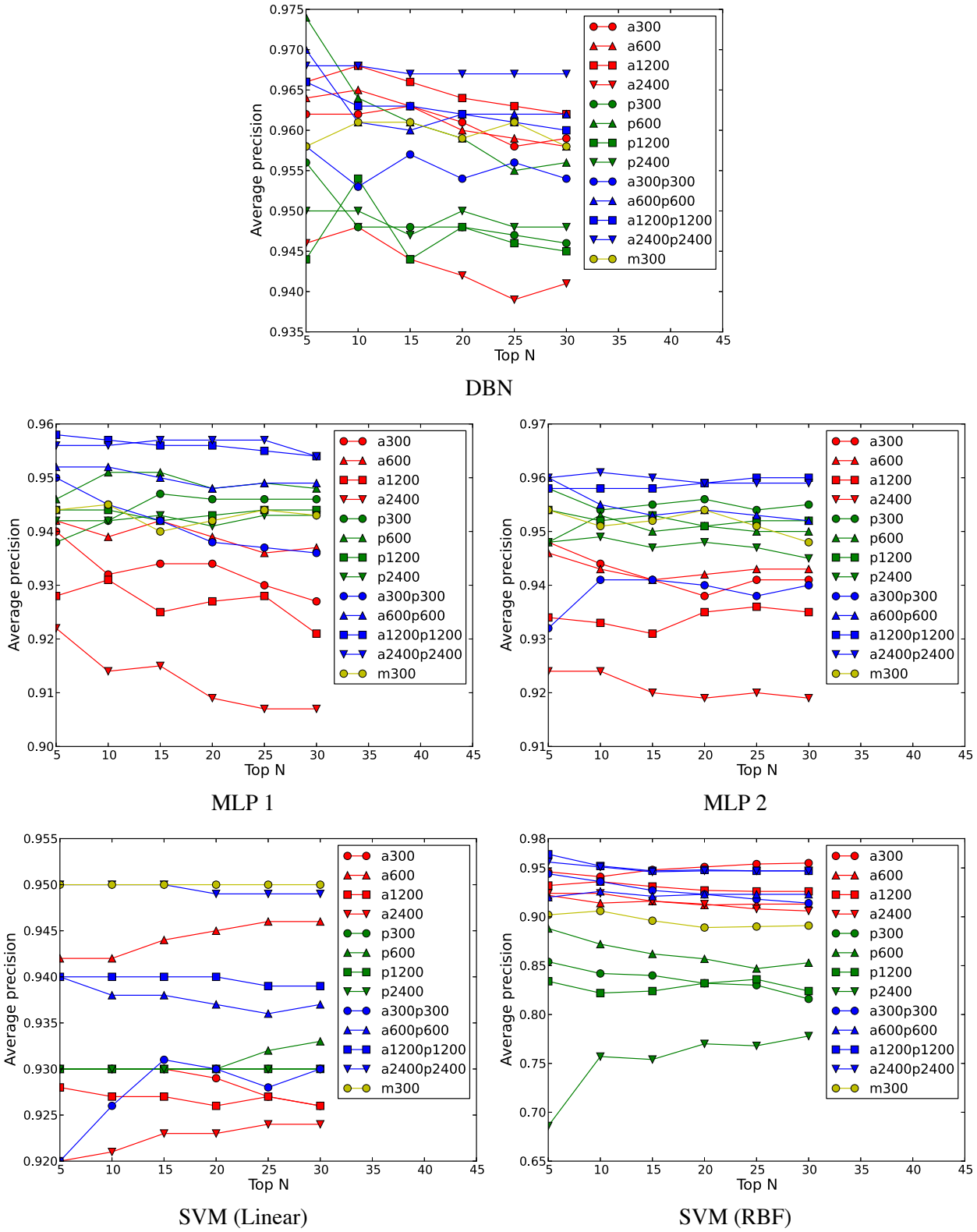
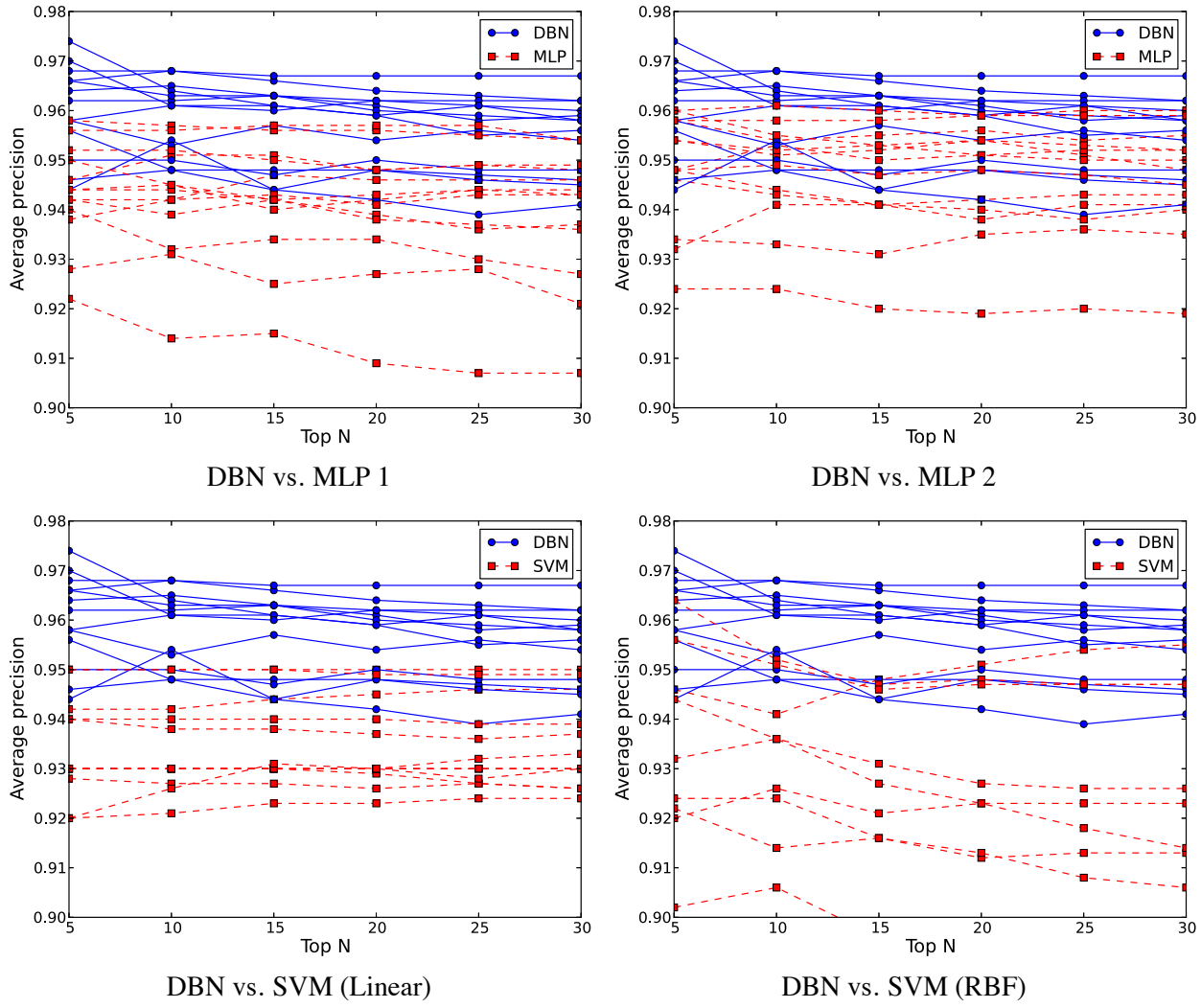


Figure 6: Average precisions of DBN, MLP, and SVM for top N varying from 5 to 30.



DBN vs. MLP 1 DBN vs. MLP 2

DBN vs. SVM (Linear) DBN vs. SVM (RBF)

Figure 7: Comparison of average precisions for top N varying from 5 to 30.

are not indicated in the DBN vs. SVM (RBF) graph because their precisions were lower than 0.9. Full results, however, are shown in Figure 6.

Table 4, 5, and 6 show the precisions of the baseline method and the average precisions of the machine learning methods for the top 5 and 10 sets of hyperparameters in ascending order of the cross-validation errors, respectively, when using different data sets for training. First, in contrast to the machine learning methods, we see that adding noisy training data (i.e., adding only the automatically gathered data or adding both the automatically gathered and the pseudo data) was not useful for the baseline method to improve the prediction precisions: on the contrary, the noisy data significantly reduced the prediction precisions. Second, in almost all cases, the precisions of the baseline method were far lower than those of all machine learning methods. Finally, we see that in almost all cases, the DBN had the highest precision (the bold figures in the tables) of all the machine learning methods.

In addition, even when only using the base data set (i.e., the manually gathered data (m300)) for training, we can conclude from Figure 6 and Table 5 and 6 that, in all cases, the precision of DBN was the highest.

5 Conclusion

We proposed methods to predict retrieval terms from the relevant/surrounding words or the descriptive texts in Japanese by using deep belief networks (DBN), one of the two typical types of deep learn-

	m300	a300	a600	a1200	a2400	p300	p600
Baseline	0.850	0.500	0.320	0.390	0.370	0.850	0.840

	p1200	p2400	a300p300	a600p600	a1200p1200	a2400p2400
Baseline	0.840	0.840	0.510	0.320	0.390	0.370

Table 4: Precisions of the baseline.

	m300	a300	a600	a1200	a2400	p300	p600
MLP 1	0.944	0.940	0.942	0.928	0.922	0.938	0.946
MLP 2	0.954	0.948	0.946	0.934	0.924	**0.958**	0.948
SVM (Linear)	0.950	0.930	0.942	0.928	0.920	0.930	0.930
SVM (RBF)	0.902	0.946	0.922	0.932	0.924	0.854	0.888
DBN	**0.958**	**0.962**	**0.964**	**0.966**	**0.946**	0.956	**0.974**

	p1200	p2400	a300p300	a600p600	a1200p1200	a2400p2400
MLP 1	0.944	0.942	0.950	0.952	0.958	0.956
MLP 2	**0.954**	0.948	0.932	0.960	0.958	0.960
SVM (Linear)	0.930	0.930	0.920	0.940	0.940	0.950
SVM (RBF)	0.834	0.686	0.944	0.920	0.964	0.956
DBN	0.944	**0.950**	**0.958**	**0.970**	**0.966**	**0.968**

Table 5: Average precisions of DBN, MLP, and SVM for top 5.

	m300	a300	a600	a1200	a2400	p300	p600
MLP 1	0.945	0.932	0.939	0.931	0.914	0.942	0.951
MLP 2	0.951	0.944	0.943	0.933	0.924	**0.954**	0.953
SVM (Linear)	0.950	0.930	0.942	0.927	0.921	0.930	0.930
SVM (RBF)	0.960	0.941	0.914	0.936	0.924	0.842	0.872
DBN	**0.961**	**0.962**	**0.965**	**0.968**	**0.948**	0.948	**0.964**

	p1200	p2400	a300p300	a600p600	a1200p1200	a2400p2400
MLP 1	0.944	0.942	0.945	0.952	0.957	0.956
MLP 2	0.952	0.949	0.941	0.955	0.958	0.961
SVM (Linear)	0.930	0.930	0.926	0.938	0.940	0.950
SVM (RBF)	0.822	0.757	0.936	0.926	0.952	0.951
DBN	**0.954**	**0.950**	**0.953**	**0.961**	**0.963**	**0.968**

Table 6: Average precisions of DBN, MLP, and SVM for top 10.

ing. To determine the effectiveness of using DBN for this task, we tested it along with baseline methods using example-based approaches and conventional machine learning methods such as MLP and SVM in comparative experiments. The data for training and testing were obtained from the Web in both manual and automatic manners. We also used automatically created pseudo data. We adopted a grid search to obtain the optimal hyperparameters of these methods by performing cross-validation on the training data. Experimental results showed that (1) using DBN has far higher prediction precisions than using the baseline methods and has higher prediction precisions than using either MLP or SVM; (2) adding automatically gathered data and pseudo data to the manually gathered data as training data further improves the prediction precisions; and (3) DBN and SVM (RBF) are able to deal with more noisier training data than MLP, i.e., the prediction precision of DBN can be improved by adding noisy

training data, but that of MLP cannot be.

In our future work, we plan to re-confirm the effectiveness of the proposed methods by scaling up the experimental data and then start developing various practical domain-specific systems that can predict suitable retrieval terms from the relevant/surrounding words or descriptive texts.

Acknowledgments

This work was supported by JSPS KAKENHI Grant Number 25330368.

References

M. Auli, M. Galley, C. Quirk, and G. Zweig. 2013. Joint Language and Translation Modeling with Recurrent Neural Networks. *EMNLP 2013*, 1044–1054.

Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. 2007. Greedy Layer-wise Training of Deep Networks. 153–160. *NIPS 2006*, 153–160.

Y. Bengio. 2009. Learning Deep Architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127.

Y. Bengio, A. Courville, and P. Vincent. 2013. Representation Learning: A Review and New Perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1798–1828.

R. Billingsley and J. Curran. 2012. Improvements to Training an RNN Parser. *COLING 2012*, 279–294.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537.

X. Glorot, A. Bordes, and Y. Bengio. 2011. Domain Adaptation for Large-Scale Sentiment Classification: A Deep Learning Approach. *ICML 2011*, 513–520.

K. Hashimoto, M. Miwa, Y. Tsuruoka, and T. Chikayama. 2013. Simple Customization of Recursive Neural Networks for Semantic Relation Classification. *EMNLP 2013*, 1372–1376.

K. M. Hermann and P. Blunsom. 2013. The Role of Syntax in Vector Space Models of Compositional Semantics. *ACL 2013*, 894–904.

G. E. Hiton, S. Osindero, and Y. Teh. 2006. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18:1527–1554.

P. S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck 2013. Learning deep structured semantic models for web search using clickthrough data. *CIKM 2013*, 2333–2338.

N. Kalchbrenner and P. Blunsom. 2013. Recurrent Continuous Translation Models. *EMNLP 2013*, 1700–1709.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. *NIPS 2012*, 1097–1105.

H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. 2009. Convolutional Deep Belief Networks for Scalable Unsupervised Learning of Hierarchical Representations. *ICML 2009*, 609-616.

L. Li and Y. Zhao, et al. 2013. Hybrid Deep Neural Network - Hidden Markov Model (DNN-HMM) Based Speech Emotion Recognition. *ACII 2013*.

L. Liu, T. Watanabe, E. Sumita and T. Zhao. 2013. Additive Neural Networks for Statistical Machine Translation. *ACL 2013*, 791–801.

T. Luong, R. Socher, and C. Manning. 2013. Better Word Representations with Recursive Neural Networks for Morphology. *ACL 2013*, 104–113.

R. Salakhutdinov and G. E. Hinton. 2009. Semantic Hashing. *International Journal of Approximate Reasoning*, 50(7): 969-978.

F. Seide, G. Li, and D. Yu. 2011. Conversational Speech Transcription Using Context-Dependent Deep Neural Networks. *INTERSPEECH 2011*, 437-440.

R. Socher, E. H. Huang, J. Pennington, A. Y. Ng, and C. D. Manning. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. 801–809. *NIPS 2011*, 801–809.

R. Socher, J. Bauer, C. D. Manning, and A. Y. Ng. 2013. Parsing with Computational Vector Grammars. *ACL 2013*, 455–465.

R. Socher, A. Perelygin, J. Y. Wu, and J. Chuang. 2013. Recursive Deep Models for Semantic Compositionality Over a Sentiment Treebank. *EMNLP 2013*, 1631–1642.

S. Srivastava, D. Hovy, and E. H. Hovy. 2013. A Walk-Based Semantically Enriched Tree Kernel Over Distributed Word Representations. *EMNLP 2013*, 1411–1416.

M. Tsubaki, K. Duh, M. Shimbo, and Y. Matsumoto. 2013. Modeling and Learning Semantic Co-Compositionality through Prototype Projections and Neural Networks. *EMNLP 2013*, 130–140.

P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. 2008. Extracting and Composing Robust Features with Denoising Autoencoders. *ICML 2008*, 1096–1103.

P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P. A. Manzagol. 2010. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *Journal of Machine Learning Research*, 11:3371–3408.

W. Y. Zou, R. Socher, D. M. Cer, and C. D. Manning. 2013. Bilingual Word Embeddings for Phrase-Based Machine Translation. *EMNLP 2013*, 1393–1398.