Retrieval Term Prediction Using Deep Learning Methods

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 methods to predict retrieval terms from relevant/surrounding words or descriptive texts in Japanese by using deep learning methods, which are implemented with stacked denoising autoencoders (SdA), as well as deep belief networks (DBN). To determine the effectiveness of using DBN and SdA for this task, we compare them with conventional machine learning methods, i.e., multi-layer perceptron (MLP) and support vector machines (SVM). We also compare their performance in case of using three regularization methods, the weight decay (L2 regularization), sparsity (L1 regularization), and dropout regularization. The experimental results show that (1) adding automatically gathered unlabeled data to the labeled data for unsupervised learning is an effective measure for improving the prediction precision, and (2) using DBN or SdA results in higher prediction precision than using SVM or MLP, whether or not regularization methods are used.

1 Introduction

Existing Web search engines have very high retrieval performance as long as the proper retrieval terms are input. However, many people, particularly children, seniors, and foreigners, have difficulty deciding on the proper retrieval terms for representing the retrieval objects,[1] especially in searches related to technical fields. Support systems are in place for search engine users that show suitable retrieval term candidates when 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. It is therefore necessary to develop various domain-specific information retrieval support systems that can predict suitable retrieval terms from relevant/surrounding words or descriptive texts in Japanese.

In recent years, on the other hand, 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 & syntax (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). How-

[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/)

ever, there have been no studies on applying deep learning to information retrieval support tasks. It is therefore necessary to confirm whether deep learning is more effective than other conventional machine learning methods in this task.

Two objectives were cited above. One was to develop an effective method for predicting suitable retrieval terms and the other was 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. On this basis, Ma et al. (2014) proposed a method to predict retrieval terms in computer-related fields using machine learning methods with deep belief networks (DBN) (Hinton et al., 2006; Lee et al., 2009; Bengio et al., 2007; Bengio, 2009; Bengio et al., 2013). In small-scale experiments they showed that using DBN resulted in higher prediction precision than using either a multi-layer perceptron (MLP) or support vector machines (SVM). To evaluate their proposed method more reliably, the first thing we must do is scale up the experiments. In general, it is not easy to obtain large training data, particularly labeled data for supervised learning. Fortunately, deep learning consists of both unsupervised learning and supervised learning, and unlabeled data can be collected relatively easily. Second, since a number of regularization methods (Srivastava et al., 2014) have been adopted for improving the generalization performance of neural networks, we also need to conduct evaluations when regularization is used.

This study is an enhanced version of the previous work of Ma et al. (2014), and the retrieval terms were confined to computer-related fields as before. We implemented deep learning not only with the DBN as done in the previous work of Ma et al. (2014), but also with stacked denoising autoencoders (SdA) (Bengio et al., 2007; Bengio, 2009; Bengio et al., 2013; Vincent et al., 2008; Vincent et al., 2010). We conducted extensive experiments in which a large amount of unlabeled data was automatically collected from the Web (as a result, the amount of data and the number of labels used in this study were about ten times larger than those used in the previous study (Ma et al., 2014)), and then we compared the performance between DBN and SdA, and between DBN/SdA and conventional machine learning methods, in the respective cases of using or not using regularization methods, i.e., weight decay (L2 regularization), sparsity (L1 regularization), and dropout regularization.

Experimental results show that using SdA achieves the highest prediction precision among all the methods and that using both DBN and SdA produces higher prediction precision than that achieved using either MLP or SVM, when regularization methods are not used. On the other hand, when regularization methods are used MLP and DBN performance is improvement in some cases, whereas no performance improvement can be found in SdA. Whether or not regularization methods are used, however, the order of superiority among SdA, DBN, and MLP remains unchanged. The experimental results also show that adding automatically gathered unlabeled data to the labeled data for unsupervised learning is an effective measure for improving the prediction precision.

2 Data

In this section, we describe how the training and testing data were obtained and how the feature vectors of the inputs were constructed from the data for machine learning.

2.1 Labeled Data

For supervised learning and testing, a labeled data set 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, so we call the retrieval terms labels here. Table 1 gives examples of these pairs, where the "Relevant/surrounding words" are those extracted from descriptive texts in accordance with the steps described in Subsection 2.3.

A total of 1,234 pieces of data labeled with 100 different labels (i.e., 1,234 pairs of inputs and labels) were manually collected from 22 computer terminology Web sites.

2.2 Unlabeled Data

Unlabeled data can be used for unsupervised learning and are obtained from the Web in an automatic manner. We respectively combine five words or

Labels (Retrieval terms)	Inputs (Descriptive texts or relevant/surrounding words; translated from Japanese)	
Graphics board	**Descriptive text**	Also known as: graphics card, graphics 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 it does not receive a command. The graphics board is the device that outputs the commands. Two types of data exist that the graphics board needs to process on the PC: 2D (planar data) and 3D (three-dimensional data).
	Relevant/surrounding words	screen, picture, eye, displays, as commanded, command, device, two types exist, data, process, on the PC, 2D, planar data, 3D, three-dimensional data.
	Descriptive text	A device that provides independent functions for outputting or inputting video as signals on a PC or various other types of computers in the form of an expansion card (expansion board). The drawing speed, resolution, and 3D performance vary according to the chip and memory mounted on the card.
	Relevant/surrounding words	independent, functions, outputting, inputting, video, signals, PC, various other types, computer, expansion card, expansion board, drawing speed, resolution, 3D performance, chip, memory, mounted, card
Main memory	**Descriptive text**	A device that stores data and programs on a computer. Also known as the 'primary memory device'. Since main memory uses semiconductor elements to electrically record information, its operation is fast and it can read and write directly to and from the central processing unit (CPU). However, it has a high cost per unit volume and so cannot be used in large quantities.
	Relevant/surrounding words	device, stores, data, programs, on a computer, primary memory device, main memory, uses, semiconductor elements, electrically, record, operation, fast, read and write directly, central processing unit, CPU, cost, per unit volume, used, in large quantities.
	Descriptive text	Main memory is a device that temporarily stores data on a PC. Increasing the volume of the main memory is important in terms of increasing PC performance.
	Relevant/surrounding words	main memory, device, temporarily, stores, data, PC, volume, performance

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

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*, "graphics board"), then the retrieval terms are グラフィックボードとは (*gurafikku boudo toha*, "what is graphics board"), グラフィックボードは (*gurafikku boudo ha*, "graphics board is"), etc.) and then use these terms to obtain the relevant Web pages by a Google search. Because data gathered in this way might have incorrect labels, i.e., labels that do not match the descriptive texts, we use them as unlabeled data. We obtained 25,000 pieces of data (i.e., inputs) in total.

2.3 Word Extraction and Feature Vector Construction

Relevant/surrounding words are extracted from descriptive texts in steps (1)–(4) below, and the inputs are represented by feature vectors in machine learning constructed in steps (1)–(6): (1) perform morphological analysis on the labeled data that are used for training and extract all nouns, including proper nouns, verbal nouns (nouns forming verbs by adding the word する (*suru*, "do")), and general nouns; (2)

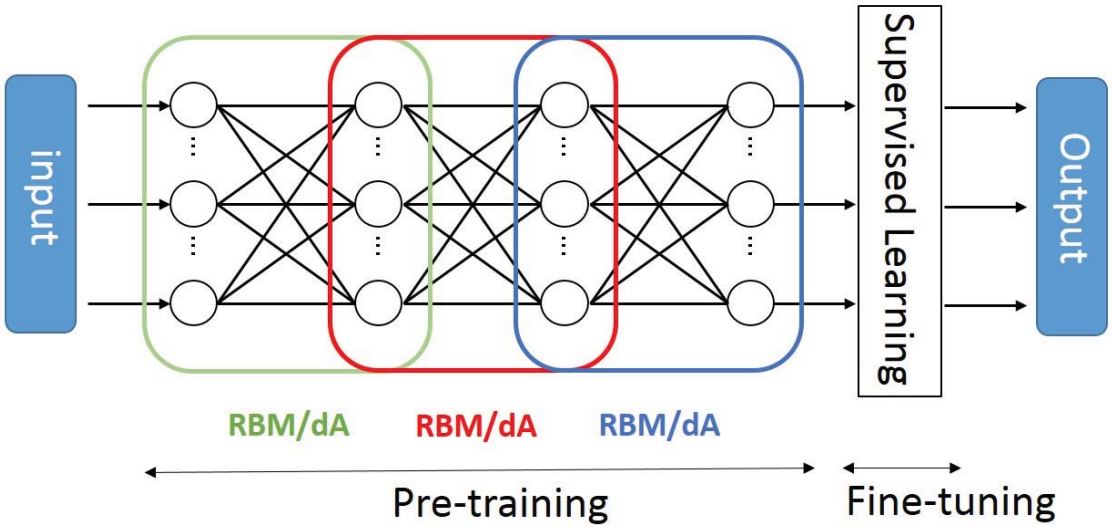

Figure 1: An example of deep neural networks consisting of DBN or SdA.

connect the nouns successively appearing as single words; (3) extract the words whose appearance frequency in each label is ranked in the top 30; (4) exclude the words appearing in the descriptive texts of more than 20 labels; (5) use the words obtained in the above steps as the vector elements with binary values, taking value 1 if a word appears and 0 if not; and (6) morphologically analyze all data described in Subsections 2.1 and 2.2, and construct the feature vectors in accordance with step (5).

3 Deep Learning and Regularization

Deep learning consists of unsupervised learning for pre-training to extract features and supervised learning for fine-tuning to output labels. Deep learning can be implemented by two typical approaches: 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 autoencoders (SdA) (Bengio et al., 2007; Bengio, 2009; Bengio et al., 2013; Vincent et al., 2008; Vincent et al., 2010). The same supervised learning method can be used with both of these approaches; i.e., both approaches can be implemented with a single-layer or multi-layer perceptron or other techniques (linear regression, logistic regression, etc.), while a different unsupervised learning method is used; i.e., a DBN is formed by stacking restricted Boltzmann machines (RBM), and an SdA is formed by stacking denoising autoencoders (dA) using a greedy layer-wise training algorithm. In this work, we use SdA as well as DBN, both of which use logistic regression for supervised learning.

Figure 1 shows an example of deep neural networks composed of three RBM or dA for pre-training and a supervised learning device for fine-tuning. Naturally the number of RBM/dA is changeable as needed. As shown in the figure, the hidden layers of the earlier RBM/dA become the visible layers of the new RBM/dA.

A number of regularization methods have been proposed to prevent overfitting and to improve the generalization performance. Weight decay (L2 regularization) is a method to prevent the weights from becoming too large by adding the sum of the squared weights to an error function, and sparsity (L1 regularization) is a method to help select features in sparse feature spaces by adding the sum of the absolute values of weights to an error function, which leads to many of the weights becoming zero. In contrast, dropout (Srivastava et al., 2014) is a method to train different models for each piece of a training data set by randomly removing units with probability p from input and hidden layers. At test time, all units are then always present (not removed) and their

Machine learning methods	Hyperparameters	Values
DBN	structure (hidden layers)	662, 992-662, **1103-882-662**[2], 1985, 1985-1985, 1985-1985-1985, 2646, 1985-2646, 1764-2205-2646
	ϵ of pre-training	0.05, 0.1, 0.5
	ϵ of fine-tuning	0.05, 0.1, 0.5
	epoch of pre-training	10, 50, 100
	stop of fine-tuning	when the training error (average of 0/1 loss) is below 0.03
SdA	structure (hidden layers)	662, 992-662, 1103-882-662, 1985, 1985-1985, 1985-1985-1985, 2646, 1985-2646, 1764-2205-2646
	ϵ of pre-training	0.05, 0.1, 0.5
	ϵ of fine-tuning	0.05, 0.1, 0.5
	epoch of pre-training	10, 50, 100
	stop of fine-tuning	when the training error (average of 0/1 loss) is below 0.03
MLP	structure (hidden layers)	662, 992-662, 1103-882-662, 1985, 1985-1985, 1985-1985-1985, 2646, 1985-2646, 1764-2205-2646
	ϵ	27 divisions between 10^{-2}-10^{0} in a logarithmic scale
	stop of training	when the training error (average of 0/1 loss) is below 0.03
SVM (Linear)	C	243 divisions between 10^{-6}-10^{6} in a logarithmic scale
SVM (RBF)	C	16 divisions between 10^{-4}-10^{4} in a logarithmic scale
	γ	15 divisions between 10^{-4}-10^{4} in a logarithmic scale
Bernoulli Naïve Bayes	additive smoothing	122 divisions between 10^{-6}-10^{0} in a logarithmic scale
	learning of prior probability	True, False

Table 2: Hyperparameters for grid search used in the comparative experiments of different training data sets and different machine learning methods without regularization.

weights are multiplied by 1-p.

[2]As an example, the structure (hidden layers) **1103-882-662**, shown as bold in the table, refers to a DBN with a 1323-**1103-882-662**-100 structure, where 1323 and 100 respectively refer to dimensions of the input and output layers. These figures were set not in an arbitrary manner. The first three structures are decreasing (pyramid-like) size and all hidden layers were set to 3/6, 4/6, 3/4, and 5/6 times smaller than that of the input layer, i.e., $662 = 1323 \times 3/6$, $882 = 1323 \times 4/6$, $992 = 1323 \times 3/4$, and $1103 = 1323 \times 5/6$. The last three structures are increasing (upside down pyramid) size and all hidden layers were set to 8/6, 9/6, 10/6, and 12/6 times larger than that of the input layer, i.e., $1764 = 1323 \times 8/6$, $1985 = 1323 \times 9/6$, $2205 = 1323 \times 10/6$, and $2646 = 1323 \times 12/6$. The middle three structures were set in accordance with the recommendations of (Bengio, 2012) that using the same size works generally as well as or better than using a decreasing (pyramid-like) or increasing (upside down pyramid) size.

4 Experiments

4.1 Experimental Setup

We used three data sets with different amounts of data (i.e., 1,134 labeled data; 1,134 labeled data + 13,000 unlabeled data; and 1,134 labeled data + 25,000 unlabeled data) for unsupervised learning, the same 1,134 labeled data for supervised learning, and the remaining 100 labeled data for testing. The

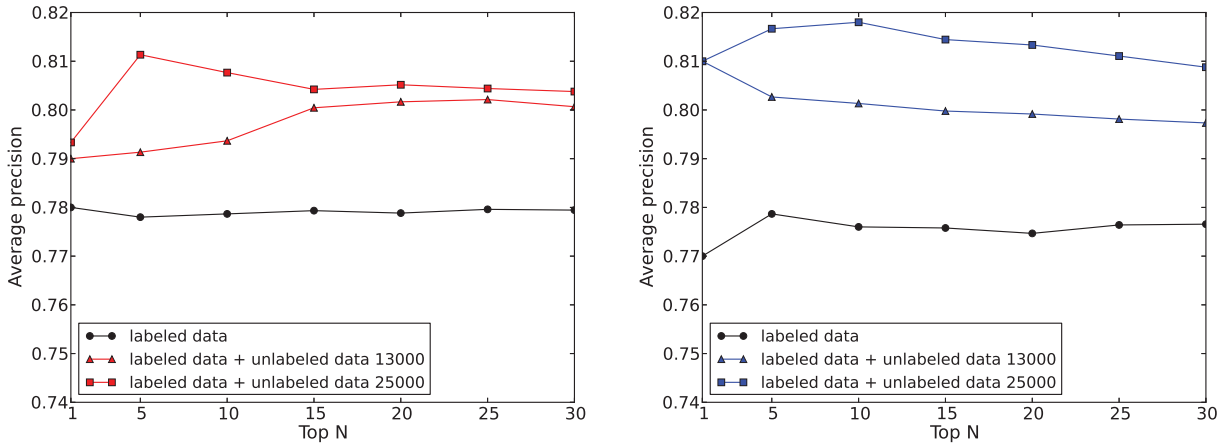

Figure 2: Average precision values obtained using different training data without regularization.

dimension of the feature vectors constructed from the 1,134 labeled training data in accordance with the steps in Subsection 2.3 was 1,323.

The optimal hyperparameters of the various machine learning methods used were determined by a grid search using 5-fold cross-validation on training data. To avoid unfair bias toward DBN/SdA during cross-validation due to DBN/SdA having more hyperparameters than MLP/SVM, we divided MLP/SVM hyperparameter grids more finely than that of the DBN/SdA so that they had the same hyperparameter combinations (hyperparameter sets) as those of DBN/SdA. Also, to avoid unfair bias toward DBN/SdA/MLP with regularization during cross-validation due to they having more hyperparameters than those without regularization, we divided DBN/SdA/MLP with no regularization hyperparameter grids more finely than those of the DBN/SdA/MLP with regularization so that they had the same hyperparameter combinations as those of DBN/SdA/MLP with regularization.

Table 2 shows the hyperparameters for grid search used in the comparative experiments of different training data sets and different machine learning methods without regularization. We therefore had 243 hyperparameter sets in total for these experiments. On the other hand, hyperparameters for grid search used in the comparative experiments with and without regularization are not shown in a table because of space limitations. In these experiments, the

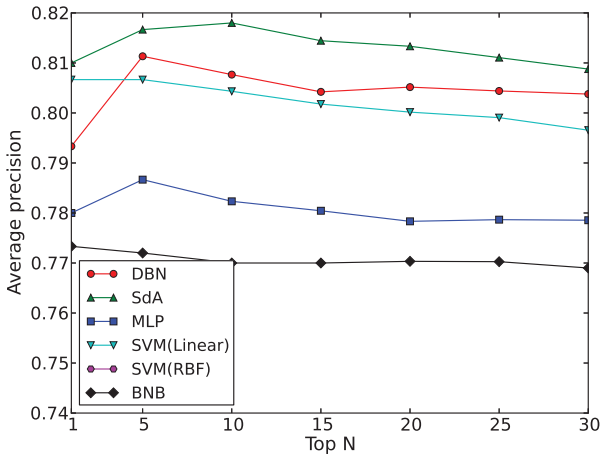

Figure 3: Average precision values obtained using different methods without regularization.

hyperparameters of DBN/SdA/MLP without regularization differ from those in Table 2. The differences were due to the above-cited measure we took to avoid unfair bias. As a result, we had 2,187 hyperparameter sets in total for these experiments.

4.2 Results

Figure 2 compares the testing data precision obtained with DBN and SdA when using different training data sets. The precision values are averages

	N=1	N=5	N=10
BNB	0.773	0.772	0.770
MLP	0.780	0.789	0.790
SVM (Linear)	0.807	0.807	0.804
SVM (RBF)	0.717	0.719	0.716
DBN	0.793	0.811	0.808
SdA	**0.810**	**0.817**	**0.818**

Table 3: Average precisions values obtained using different methods without regularization.

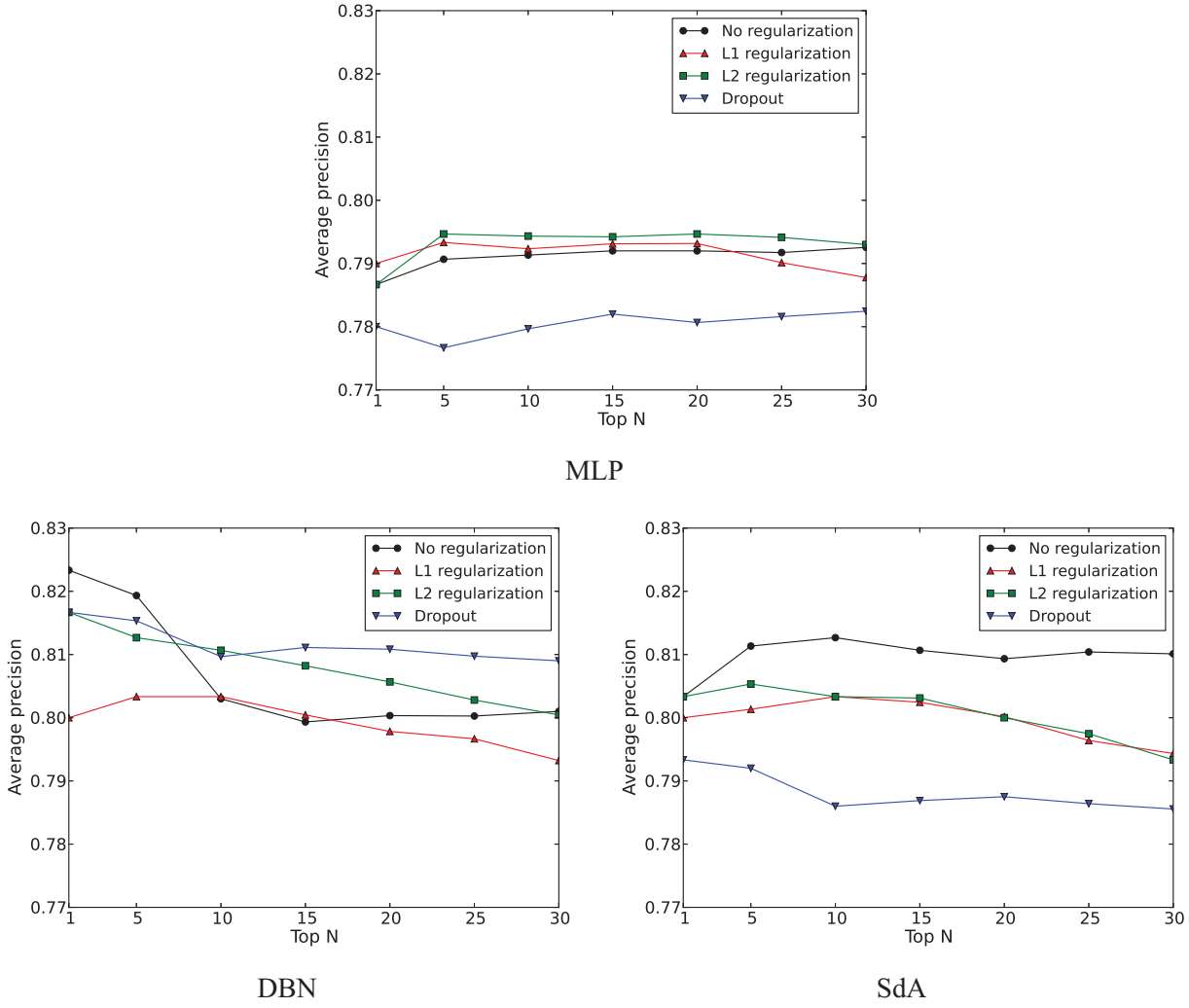

Figure 4: Comparison of average precision values obtained with and without regularization.

when using the top N sets of the hyperparameters in ascending order of the cross-validation errors, with N varying from 1 to 30. As shown in the figure, the precision of both DBN and SdA can be improved by adding the unlabeled data to the labeled data as training data, and both DBN and SdA have higher precision when using a larger amount of unlabeled data.

Figure 3 compares the testing data precision values obtained when using the largest data set (1,134 labeled data + 25,000 unlabeled data) for unsupervised learning, when using different learning meth-

	N=1	N=5	N=10
BNB	0.773	0.773	0.773
MLP	0.787	0.791	0.791
MLP with L1	0.790	0.793	0.792
MLP with L2	0.787	0.795	0.794
MLP with Dropout	0.780	0.777	0.780
SVM (Linear)	0.807	0.809	0.807
SVM (RBF)	0.730	0.737	0.735
DBN	**0.823**	**0.819**	0.803
DBN with L1	0.800	0.803	0.803
DBN with L2	0.817	0.813	0.811
DBN with Dropout	0.817	0.815	0.810
SdA	0.803	0.811	**0.813**
SdA with L1	0.800	0.801	0.803
SdA with L2	0.803	0.805	0.803
SdA with Dropout	0.793	0.792	0.789

Table 4: Average precision values obtained using different methods with and without regularization.

ods and Bernoulli Naïve Bayes (BNB), which is used as a baseline. We can see at a glance from the figure that the performance of SdA is superior to that of DBN and that both DBN and SdA are generally superior to BNB, MLP, and SVM. We should point out that the results for SVM (RBF) are not indicated in the figure because the precision values were lower than 0.74. Table 3 lists the specific average precision values obtained using different learning methods when N=1, 5, and 10.

Figure 4 and Table 4 compare the testing data precision values for MLP, DBN, and SdA with and without regularization[3]. The figure and table show that the performance of MLP and DBN improved in some cases by using regularization, whereas no performance improvement can be found for SdA. However, both DBN and SdA outperformed BNB, MLP and SVM whether regularization was used or not.

5 Conclusion

We presented methods to predict retrieval terms from relevant/surrounding words or descriptive texts in Japanese by using deep belief networks (DBN) and stacked denoising autoencoders (SdA). Experimental results based on a relatively large scale confirmed that (1) adding automatically gathered unlabeled data to the labeled data for unsupervised learning was an effective measure for improving the prediction precision, and (2) using either DBN or SdA definitely achieved higher prediction precision than that obtained using multi-layer perceptron (MLP), whether weight decay (L2 regularization), sparsity (L1 regularization), or dropout regularization was used. Both DBN and SdA achieved higher precision than Bernoulli Naïve Bayes (BNB) and support vector machines (SVM).

In the future, we plan to start developing various practical domain-specific systems that can predict suitable retrieval terms from the relevant/surrounding words or descriptive texts.

[3]It should be noted that the precision values obtained without regularization (shown in Figure 4 and Table 4) differ from those shown in Figure 3 and Table 3. This is because different numbers of hyperparameter sets were used for grid searching between the two experiments as described in Subsection 4.1.

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. 2012. Practical Recommendations for Gradient-Based Training of Deep Architectures. *eprint arXiv1206.5533*, 1–33.

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*,

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.

Q. Ma, I. Tanigawa, and M. Murata. 2014. Retrieval Term Prediction Using Deep Belief Networks. *Paclic 2014*.

S. J. Nowlan and G. Hinton. 1992. Simplifying Neural Networks by Soft Weight-sharing. *Neural Computation*, 4:474-493.

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. *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.

N. Srivastava, G. Hinton, A. Krizhevsky, H. Sutskever, and R. Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

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.