Unsupervised Domain Adaptation for Word Sense Disambiguation using Stacked Denoising Autoencoder

Kazuhei Kouno, Hiroyuki Shinnou, Minoru Sasaki, Kanako Komiya
Ibaraki University, Department of Computer and Information Sciences
4-12-1 Nakanarusawa, Hitachi, Ibaraki JAPAN 316-8511
15nm707g@vc.ibaraki.ac.jp
{hiroyuki.shinnou.0828, minoru.sasaki.01, kanako.komiya.nlp}
@vc.ibaraki.ac.jp

Abstract

In this paper, we propose an unsupervised domain adaptation for Word Sense Disambiguation (WSD) using Stacked Denoising Autoencoder (SdA). SdA is an unsupervised learning method of obtaining the abstract feature set of input data using Neural Network. The abstract feature set absorbs the difference of domains, and thus SdA can solve a problem of domain adaptation. However, SdA does not always cope with any problems of domain adaptation. Especially, difficulty of domain adaptation for WSD depends on the combination of a source domain, a target domain and a target word. As a result, any method of domain adaptation for WSD has adverse effect for a part of the problem, Therefore, we defined the similarity between two domains, and judge whether we use SdA or not through this similarity. This approach avoids an adverse effect of SdA. In the experiments, we have used three domains from the Balanced Corpus of Contemporary Written Japanese and 16 target words. In comparison with baseline, our method has got higher average accuracies for all combinations of two domains. Furthermore, we have obtained better results against conventional domain adaptation methods.

1 Introduction

In this paper, we propose an unsupervised method of domain adaptation for Word Sense Disambiguation (WSD) using Stacked Denoising Autoencoder (SdA).

WSD is the task of identifying the sense of a target word in a sentence. In general, supervised learning, such as Support Vector Machine (SVM), can be used for this task because of the fact that this approach is highly accurate. However, if the training and test data come from different domains, the accuracy of this approach is lowered. This problem is called a *domain adaptation* (Søgaard, 2013). It is considered that this problem occurs due to the difference between the distributions of features in training and test data.

SdA is an unsupervised learning method of obtaining the abstract feature of the input data (basic feature) using Neural Network (Vincent et al., 2010). Recently it has been shown that a higher accuracy in voice and character recognition has been obtained using SdA (Le et al., 2012). We have applied this method to a domain adaptation for WSD and have shown that the abstract feature obtained through SdA can avoid the problem of domain adaptation.

It is well-known from previous works that the most efficient methods for domain adaptation for WSD depend on the combination of training data (from the source domain) and test data (from the target domain) (Komiya and Okumura, 2011) (Komiya and Okumura, 2012). Furthermore, in an unsupervised domain adaptation method, even if the accuracy is improved in the combination of the source and target domains, the accuracy rate hardly improve. As a result, the accuracy rate on average of the method decreases, or remains the same. In other words, there are accuracy limitations with each method. In our method, we choose whether or not to apply SdA based on the similarity of features. Our method cannot be applied in the case for pair of do-

29th Pacific Asia Conference on Language, Information and Computation: Posters, pages 224 - 231
Shanghai, China, October 30 - November 1, 2015

mains as they are not suitable for SdA.

In our experiment, we have used three domains: Yahoo! Answers (OC), Books (PB), and newspaper (PN) from the Balanced Corpus of Contemporary Written Japanese (Maekawa, 2007), along with 16 selected ambiguous words. Domain adaptation has the following six transitions: (1) PB → OC, (2) OC → PB, (3) OC → PN, (4) PN → OC, (5) PB → PN and (6) PN → PB. First, in every domain adaptation, we have compared the accuracy of the basic feature and abstract feature by SdA using SVM. As a result, SdA have been effective in half of the case of domain adaptations. Furthermore, we have explored situations when to apply SdA or not. Consequently, the SdA with similarity of features is effective in all domain adaptations.

2 Domain Adaptation for WSD

Frequently, the word has multiple senses. Word Sense Disambiguation (WSD) is the task of identifying a sense of the such word in a sentence.

In general, supervised learning like SVM can be used for this task, because this approach shows a high accuracy. However, in these methods, training and test data must come from same domain. In the case of WSD, these are often obtained from different domains. For example, to learn the classifier using sentences from books as training data, and then classify the word in the sentence from newspaper. In this case, it can't well identify the test data from newspaper (target domain) by the classifier which is learned by books (source domain). To solve this problem, tuning the classifier that is learned by training data from source domain to match the test data from target domain is necessary. It is called *domain adaptation* (Søgaard, 2013).

It is considered that this problem occurs from the difference between distributions of features in training and test data. Therefore, we attempt to absorb it by SdA.

3 Related Work

Inductive learning is used not only WSD but also many natural language processing tasks, and domain adaptation problem will occur. There are two types of methods for this problem. One is a supervised domain adaptation using labeled data in the target domain and the other is an unsupervised domain adaptation that does not use it. Typically, in the domain adaptation tasks, supervised and semi-supervised learning show the high accurate (Chapelle et al., 2006). However, supervised learning are inappropriate in WSD because they use labeled data of target domain, even though the data of target domain is new data. Although semi-supervised learning requires many data of target domain, the data for each target word is not so many in WSD. Therefore, unsupervised learning is appropriate in domain adaptation for WSD. SdA for use in this study is an unsupervised learning method, and our method can be classified into unsupervised domain adaptation.

As research on unsupervised domain adaptation, there are structural correspondence learning (SCL) (Blitzer et al., 2006) and learning under covariate shift (Sugiyama and Kawanabe, 2011). In SCL, measure the mutual information from label and feature value; features the value is large are elected to Pivot feature. Features to co-occur with Pivot feature are used for classification. This is based on the idea that Pivot features are different depending on the domain, in contrast, feature to co-occur with Pivot feature are effective in classification. Learning under covariate shift is regarded as weighted learning, where sentence x is weighted with the probability density ratio $w(x) = P_T(x)/P_S(x)$. There are many methods to calculate probability density ratio. In this paper, we adopt unconstrained least squares importance fitting (uLSIF) (Kanamori et al., 2009) because it shows good performance and quick calculation time.

These approaches depend on the combination of source and target domain; there is also case that accuracy is going to negative. As a result, accuracy rate has been decreased, or dose not develop on average.

4 Stacked Denoising Autoencoder

SdA is an unsupervised learning method of obtaining the abstract feature of input data (basic feature) by using Neural Network. SdA is composed of multiple Denoising Autoencoder (dA). As mentioned above, domain adaptation for WSD has a problem that the accuracy is lowered from the difference between distributions of features in training and test

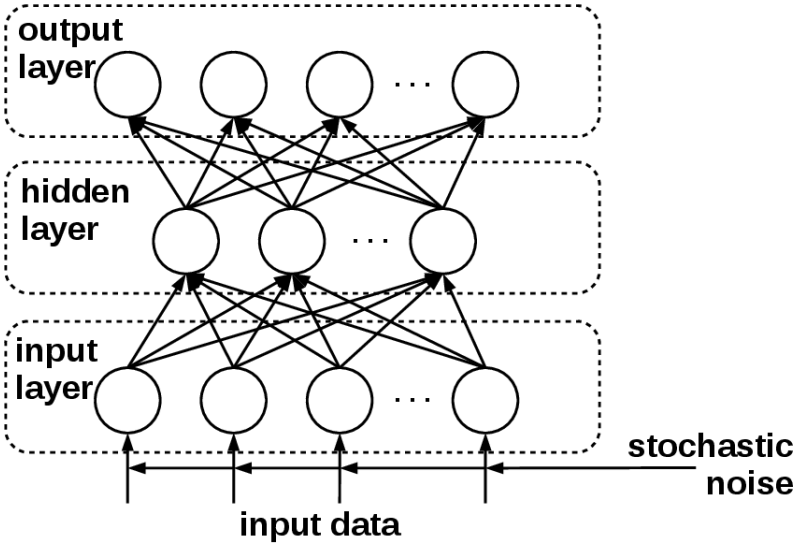

Figure 1: Denoising Autoencoder

data. The abstract feature obtained through SdA can avoid the problem of domain adaptation.

4.1 Denoising Autoencoder

The dA has input layer, hidden layer and output layer, as shown in Figure 1. At first, append stochastic noise to the input data and transmit to the input layer. Then, the data on the input layer is encoded and transmitted to the hidden layer. Similarly, data on the hidden layer is decoded and transmitted to the output layer. In this model, to learn the encoder and decoder, such as error of input data (without noise) and output layer becomes smaller. In other words, dA learns the model, such as to eliminate the noise that was added at first.

Number of nodes in the input and output layer are equal to the dimensions of input data. Typically, number of nodes in the hidden layer is set to be smaller than other layers. If the input data $x = \{x_1, x_2, \cdots, x_N\}$ and input layer with noise $\tilde{x}$, mapping from the input layer $\tilde{x}$ to the hidden layer y, and from the hidden layer y to the output layer z are represented by the following formula y, z.

$$y = \sigma(W\tilde{x} + b)$$

$$z = \sigma(W^{\mathrm{T}}y + b')$$

where b, b', W and W^{T} indicate bias, another bias, the weight matrix and the transposed matrix of W respectively. The $\sigma(\cdot)$ indicates sigmoid function:

$$\sigma(x) = \frac{1}{1 + e^{-x}}.$$

Finding the W (or W^{T}), b and b', such as mean squared error is minimized using a Stochastic Gradient Descent(SGD). Hidden layer y obtained in this process is the abstract feature of the input data x, because it can be restored the input data by decoder; nevertheless number of nodes is less.

4.2 Stacked Denoising Autoencoder

SdA is a model of stacked multiple dA, as shown in Figure 2. At first, to learn using dA that the input is the input data (call dA_1). Then, to learn using dA that the input is hidden layer of dA_1(call dA_2). In dA_3, the input is hidden layer of dA_2; SdA stacks learning by repeating this process. In this way, the abstract feature is gradually obtained from the input data. Note that output layers for each dA are used only to calculate the mean squared error; mainly, hidden layers are used on SdA.

In this paper, connecting input data and the abstract feature, to absorb the difference in distributions of features between training and test data.

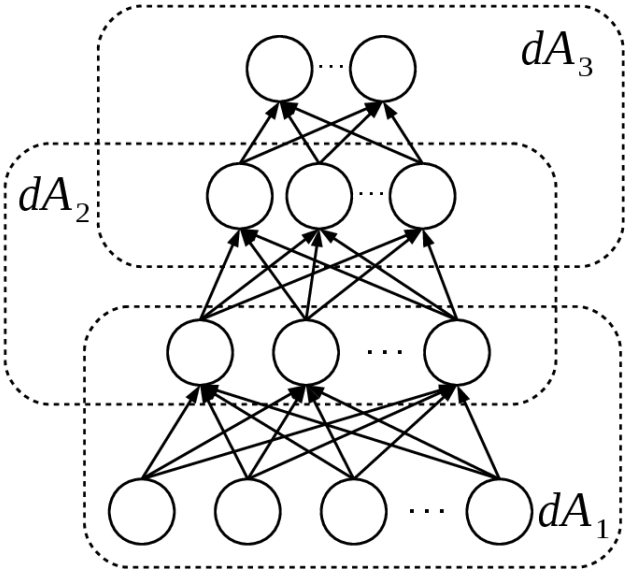

Figure 2: Stacked Denoising Autoencoder

Specifically, to extract n_{abst}-dimensional abstract feature x_{abst} from n-dimensional input data x by SdA, and then, x and x_{abst} are separately normalized. We use the data that x and x_{abst} are connected to classification by SVM.

5 Similarity of feature

In chapter 3, we introduced previous studies using SCL or uLSIF, as unsupervised domain adaptation. These approaches depend on the combination of source and target domain and there are also cases that accuracy is going to negative. As a result, accuracy rate has been decreased, or dose not develop on average. In other words, the best method for domain adaptation for WSD depends on the combination of source and target domain. Therefore, we choose whether to apply SdA based on the combination of source domain, target domain and a target word.

Configuring small number of nodes in the hidden layer than the other layers; SdA reduces dimension of data. SdA is expected that to project distributions of features from source and target domain. If both training and test data have little commonality, SdA requires a lot of data to learn a model. Typically, it is not possible to learn a better model, since training and test data are less in WSD. Therefore, to calculate similarity of feature, and then apply the SdA if this value is large.

While cosine similarity and mutual information are typical as a way to measure the similarity, we use simple approach that calculate the ratio of the number of common dimensions to all dimensions. Specifically, to determine occurrence vector of dimensions $\vec{S}$ and $\vec{T}$ for the training data X_S and test data X_T, and then to calculate the similarity P_f by following equations:

$$P_f = \frac{\vec{T} \cdot \vec{S}}{n}$$

dimension of training data X_S and test data X_T are $dim\vec{S}$ and $dim\vec{T}$, respectively. Where $dim\vec{S} = dim\vec{T}$ is satisfied; there are represented as n. If P_f is greater than the threshold T, it is regarded that training and test data have some commonality, and then apply SdA.

Table 1: Target words

word	dictionary # of senses	OC freq. of word	OC # of senses	PB freq. of word	PB # of senses	PN freq. of word	PN # of senses
iu(言う)	3	666	2	1114	2	363	2
ireru(入れる)	3	73	2	56	3	32	2
kaku(書く)	2	99	2	62	2	27	2
kiku(聞く)	3	124	2	123	2	52	2
kodomo(子供)	2	77	2	93	2	29	2
jikan(時間)	4	53	2	74	2	59	2
jibun(自分)	2	128	2	308	2	71	2
deru(出る)	3	131	3	152	3	89	3
toru(取る)	8	61	7	81	7	43	7
baai(場合)	2	126	2	137	2	73	2
hairu(入る)	3	68	4	118	4	65	3
mae(前)	3	105	3	160	2	106	4
miru(見る)	6	262	5	273	6	87	3
motsu(持つ)	4	62	4	153	3	59	3
yaru(やる)	5	117	3	156	4	27	2
yuku(ゆく)	2	219	2	133	2	27	2
average	3.44	148.19	2.94	199.56	3.00	75.56	2.69

6 Experiment

6.1 Data and Methods

In the experiment, we compare the effect by following methods:

- baseline: classify the basic feature by SVM
- uLSIF
- SCL
- SdA
- proposed method：SdA using similarity

We use the data from the Balanced Corpus of Contemporary Written Japanese (BCCWJ (Maekawa, 2007)) that has word sense tags by a Japanese WSD SemEval-2 task (Okumura et al., 2010). Among them, we use three domains as different domains: Yahoo! Answers (OC), Books (PB) and Newspaper (PN). Table 1 indicates information of the target word, the number of senses registered in the dictionary, and the number of senses and the frequency in each corpus [1]. All methods learn the classifier using the traing data from source domain; and then, classify the test data from target domain by the classifier (as represented by *source* → *target*). There are six domain adaptation patterns: (1) PB → OC, (2) OC → PB, (3) OC → PN, (4) PN → OC, (5) PB → PN and (6) PN → PB. There are six domain adaptations and sixteen target words; the experiments are made 96 ways. We evaluated each methods by following. First, to calculate the accuracy rate for each combination of source domain, target domain and target word. Then, to calculate the average for each domain adaptation. Similarly, to calculate average of 96 pairs; they are accuracy of each methods. In the proposed method, threshold T of similarity is equal to 0.2; if $P_f > 0.2$, then we choose to apply SdA.

In this experiment, we use 8 kinds of features for a sentence, that is an instance. They are shown in Table 2, where w and w_i represent target word and the i-th word from the word w respectively.

[1]The word "入る (hairu)" has three senses in the dictionary, but there are four senses in OC and PB. This is because our used sense tagged corpus accepts new senses.

Table 3: Average accuracy rate (%)

Domain Adaptation	baseline	uLSIF	SCL	SdA	our method
OC → PB	71.33	71.34	71.34	71.09	**71.43**
PB → OC	70.10	70.45	70.18	71.01	**70.93**
OC → PN	68.81	68.98	**69.24**	68.18	68.81
PN → OC	69.09	69.05	68.94	67.49	**69.24**
PB → PN	76.76	76.99	76.65	**77.33**	77.02
PN → PB	74.55	74.50	73.47	**75.37**	74.59
average	71.77	71.89	71.64	71.74	**72.00**

Table 2: feature of sentence

feature	content
(e0)	written of w
(e1)	parse of w
(e2)	written of w_{-1}
(e3)	parse of w_{-1}
(e4)	written of w_1
(e5)	parse of w_1
(e6)	written of independent word between w_{-3} and w_3
(e7)	Number from classification vocabulary table of *e6* (4 and 5-digit)

6.2 Parameters of SdA

We use Pylearn2[2] for learning the model of SdA. The number of repetitions of dA is twice. In dA_1 (Input is input data.), when the dimension of the input data is N, the number of nodes of hidden layer is $2/3 \times N$. In dA_2 (Input is hidden layer of dA_1), the number of nodes of hidden layer is equal to input layer's, that is following equation:

$$\begin{aligned} DimOfInput &= InputLayerOf\, dA_1 \\ &= \frac{2}{3} \times HiddenLayerOf\, dA_1 \\ &= \frac{2}{3} \times InputLayerOf\, dA_2 \\ &= \frac{2}{3} \times HiddenLayerOf\, dA_2 \end{aligned}$$

where as stated above, the number of nodes in output layer are equal to input layer's. On this calculation, round the result to an integer.

The hidden layer of dA_2 are connected to the basic feature, and then classified using SVM. Where basic and abstract feature are respectively normalized before connection. We use libsvm[3] as classification by SVM; kernel function is linear kernel that is often used in natural language processing tasks. Similarly, baseline also uses libsvm with linear kernel.

6.3 Results

Table 3 shows the result of our experiments.

In uLSIF, accuracy are improved in four domain adaptations, and on average, it's above the baseline. However, it was opposite effect for two domain adaptations. SCL and SdA also has good and bad results. Consequently, three methods were not much different. Meanwhile, proposed method showed high accuracy in five domain adaptations; there was no bad result in all domain adaptations. As a result, our proposed method shows best accuracy among all methods.

7 Discussions

In each domain adaptation, method that showed the best accuracy among the four methods baseline, uLSIF, SCL and SdA are shown in Table 4. The best method is different depending on the domain adaptation. Moreover, baseline showed the best result in PN → OC. This results suggest effectiveness of selecting the method by any way.

[2] http://deeplearning.net/software/pylearn2/

[3] http://www.csie.ntu.edu.tw/˜cjlin/libsvm/

Table 4: Best method for each domain adaptation

Domain Adaptation	Method
OC → PB	uLSIF, SCL
PB → OC	SdA
OC → PN	SCL
PN → OC	baseline
PB → PN	SdA
PN → PB	SdA

In this study, we bring in similarity of features, and choose whether to apply the SdA depending on the combination of training data, test data and target word. As a results, accuracy has improved in five domain adaptations, compared with baseline. In the other one domain adaptation, it shows improvement in the third decimal place. Our method showed a better result than the other four methods on average.

However, our method has a problem to be solved. Proposed method chooses either baseline or SdA for each combination of source domain, target domain and target word. If the pair is improved by SdA that does not use similarityP_f, improvement has decreased in our method compared to SdA. If our method rise to the same level as SdA in these pairs, it can be expected to more improve on average. The following two methods will be considered to achieve it.

1. Decreasing the threshold T.

2. If P_f is less than T, to modify parameter of SdA.

In approach 1, selectivity of SdA is increased by decreasing T. As a result, we expect that proposed method is close to the accuracy of SdA. However, if T is extremely low, the proposed method will show the same results as SdA. In the previous experiments, the T is equal to 0.2. There are experiments that the T is lowered to 0.18. The results are shown in Table 5.

Out of three domain adaptation (PB → OC, PB → PN and PN → PB) that is impaired with the our method compared to SdA, accuracy has improved in two domain adaptation (PB → PN and PN → PB) by lowering T. Moreover, it shows better results on PN → PB than the SdA. However, it is worse on PB → OC than the case of $T = 0.2$. About these results, we consider the influence of decision to apply the SdA for each pair of word and domains. Besides, in the case of $T = 0.18$, two domain adaptation have a poor accuracy as compared to baseline. Nevertheless, the method which is 0.18 shows the best results on average. For this reason, it is necessary to determine the appropriate threshold T.

In approach 2, if the similarity of feature P_f is fewer than the threshold T, we modify parameter of SdA. Consequently, SdA will get the feature close to the basic feature; the result is close to SdA. The parameter to be adjusted include the number of nodes in hidden layer for each dA, and the number of repetitions of dA. If the number of nodes in the hidden layer is increased, there is no difference between the dimensions of basic feature and abstract feature; SdA gets abstract features similar to basic feature. If, however, the number of nodes in hidden layer is large, learning requires a long time, because the bonds between each nodes are increased. Furthermore, learning data is not so much in WSD, there is not enough learning. An approach of increasing the number of repetitions of dA has also same problems, because the first dA have to set the large number of nodes. For this reason, if we have enough data and times, this approach is effective.

8 Conclusions

In this paper, we have proposed an unsupervised method of domain adaptation for WSD using SdA. Specifically, the basic features are converted to abstract features by SdA, and then, these are classified by SVM.

In the domain adaptation methods for WSD, the most powerful method is different from each other depending on the pair of source and target domains; there are also accuracy limitations within each method. In this paper, we have introduced a similarity of the features and the option of choosing whether to apply SdA or not.

In our experiments, we chose three domains and 16 selected ambiguous words. While uLSIF, SCL and SdA have shown poor accuracy in some case of domain adaptation, our method has been a better accuracy in all situations of domain adaptation and

Table 5: Average accuracy rate on additional experiment (%)

Domain Adaptation	baseline	SdA	our method ($T = 0.2$)	our method ($T = 0.18$)
OC → PB	71.33	71.09	**71.43**	71.31
PB → OC	70.10	71.01	**70.93**	70.66
OC → PN	68.81	68.18	68.81	**68.91**
PN → OC	69.09	67.49	**69.24**	68.85
PB → PN	76.76	**77.33**	77.02	77.12
PN → PB	74.55	75.37	74.59	**76.02**
average	71.77	71.74	72.00	**72.14**

had a better result as compared with other methods. In our future work, we plan to examine pair of domains where our method has not performed well as compared with SdA that dose not use similarity.

References

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain Adaptation with Structural Correspondence Learning. In *EMNLP-2006*, pages 120–128.

Olivier Chapelle, Bernhard Schölkopf, Alexander Zien, et al. 2006. *Semi-supervised learning*, volume 2. MIT press Cambridge.

Takafumi Kanamori, Shohei Hido, and Masashi Sugiyama. 2009. A Least-Squares Approach to Direct Importance Estimation. *The Journal of Machine Learning Research*, 10:1391–1445.

Kanako Komiya and Manabu Okumura. 2011. Automatic Determination of a Domain Adaptation Method for Word Sense Disambiguation using Decision Tree Learning. In *IJCNLP-2011*, pages 1107–1115.

Kanako Komiya and Manabu Okumura. 2012. Automatic Domain Adaptation for Word Sense Disambiguation Based on Comparison of Multiple Classifiers. In *PACLIC-2012*, pages 75–85.

Quoc Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Kai Chen, Greg Corrado, Jeff Dean, and Andrew Ng. 2012. Building high-level features using large scale unsupervised learning. In *ICML-2012*.

Kikuo Maekawa. 2007. Design of a Balanced Corpus of Contemporary Written Japanese. In *Symposium on Large-Scale Knowledge Resources (LKR2007)*, pages 55–58.

Manabu Okumura, Kiyoaki Shirai, Kanako Komiya, and Hikaru Yokono. 2010. SemEval-2010 Task: Japanese WSD. In *The 5th International Workshop on Semantic Evaluation*, pages 69–74.

Anders Søgaard. 2013. *Semi-Supervised Learning and Domain Adaptation in Natural Language Processing*. Morgan & Claypool.

Masashi Sugiyama and Motoaki Kawanabe. 2011. *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation* . MIT Press.

Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, and Pierre-Antoine Manzagol. 2010. Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion. *The Journal of Machine Learning Research*, 11:3371–3408.