Deep Neural Network Self-training Based on Unsupervised Learning and Dropout

Hye-Woo Lee¹, Noo-ri Kim², and Jee-Hyong Lee²

¹KT R&D Center, KT, Seoul, Korea

²Department of Electrical and Computer Engineering, Sungkyunkwan University, Suwon, Korea

Ijfis

Abstract

In supervised learning methods, a large amount of labeled data is necessary to find reliable classification boundaries to train a classifier. However, it is hard to obtain a large amount of labeled data in practice and it is time-consuming with a lot of cost to obtain labels of data. Although unlabeled data is comparatively plentiful than labeled data, most of supervised learning methods are not designed to exploit unlabeled data. Self-training is one of the semi-supervised learning methods that alternatively repeat training a base classifier and labeling unlabeled data in training set. Most self-training methods have adopted confidence measures to select confidently labeled examples because high-confidence usually implies low error. A major difficulty of self-training is the error amplification. If a classifier misclassifies some examples and the misclassified examples are included in the labeled training set, the next classifier may learn improper classification boundaries and generate more misclassified examples. Since base classifiers are built with small labeled dataset and are hard to earn good generalization performance due to the small labeled dataset. Although improving training procedure and the performance of classifiers, error occurrence is inevitable, so corrections of self-labeled data are necessary to avoid error amplification in the following classifiers. In this paper, we propose a deep neural network based approach for alleviating the problems of self-training by combining schemes: pre-training, dropout and error forgetting. By applying combinations of these schemes to various dataset, a trained classifier using our approach shows improved performance than trained classifier using common self-training.

Keywords: Neural network, Pre-training, Semi-supervised learning, Self-training, Dropout

Received: Mar. 7, 2017
Revised : Mar. 20, 2017
Accepted: Mar. 22, 2017

Correspondence to: Jee-Hyong Lee
(john@skku.edu)
©The Korean Institute of Intelligent Systems

©This is an Open Access article distributed under the terms of the Creative Commons Attribution Non-Commercial License (<http://creativecommons.org/licenses/by-nc/3.0/>) which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited.

1. Introduction

In supervised learning methods, a large amount of labeled data is necessary to find reliable classification boundaries (to train a classifier). However, it is hard to obtain a large amount of labeled data in practice and it is time-consuming with a lot of cost to obtain labels of data. Although unlabeled data is comparatively plentiful than labeled data, most of supervised learning methods are not designed to exploit unlabeled data.

Semi-supervised learning is machine learning approaches concerned with how to take advantage of unlabeled data in supervised learning [1]. The goal of semi-supervised learning methods is to find classification boundaries with a small amount of labeled data and a large amount of unlabeled data, comparable to classification boundaries found by supervised learning with a large amount of labeled data.

Self-training is one of the semi-supervised learning methods that alternatively repeat training a base classifier and labeling unlabeled data in training set. First, the classifier is trained using the given labeled training set and predicts labels of unlabeled examples. Then, confidently labeled examples are selected and included in the labeled training set. After then, the classifier is re-trained using the replenished labeled training set and predicts labels of the remaining unlabeled examples. These procedures are alternately repeated, until the size of labeled data is larger than a threshold or no further labeled data is replenished.

Most of self-training methods have adopted confidence measures to select confidently labeled examples because high-confidence usually implies low error. For example, a self-training decision tree [2] has used class proportion of leaf as confidence measure. A self-training support vector machine (SVM) [3] have used distance between examples and the classification boundary. A self-training neural network [4] has used softmax output regarding as class probability estimation.

A major difficulty of self-training is the error amplification. If a classifier misclassifies some examples and the misclassified examples are included in the labeled training set, the next classifier may learn improper classification boundaries and generate more misclassified examples. Training data with more misclassified examples will create more erroneous classifiers in the next iteration. As self-training proceeds, the classification error may be amplified.

Since base classifiers are built with small labeled dataset and are hard to earn good generalization performance due to the small labeled dataset, their accuracy is relatively low. Even examples labeled with high confidence can be misclassified because base classifiers are possibly over-fitted to the small labeled dataset. So, we need an approach which can help to construct more accurate and generalized classifiers. Even though we have an accurate classifier, mis-classification is inevitable. As mentions, misclassified examples in the labeled training set can amplify the error of the following classifier. The following classifier will learn the patterns in more misclassified examples and produces more erroneous results. In order to improve the performance of self-training methods, we need an approach to cope with this.

In this paper, we propose a deep neural network based approach for self-training by combining pre-training, dropout and error forgetting. First, we need an accurate and generalized base classifier to improve the performance of self-training with small initial training data. One way of improving generalization performance of classifiers is utilizing unlabeled data in

the training procedure to learn the latent information in whole training data. Recently, some pre-training methods using unlabeled data are developed for deep neural network [5]. These pre-training methods guide learning of the randomly initialized neural network to basins of attraction of minima that support better generalization [6]. With the pre-training with unlabeled data, neural networks can be more accurate and generalized even with small labeled data.

Although deep neural networks are pre-trained, there is a possibility of over-fitting to the small initial labeled data. If the amount of initial labeled training data is too small, the trained classifier tends to learn unnecessary details (e.g., observation noise) in the labeled training data. Therefore, a way to regularize over-fitting is necessary. In this paper, dropout, which is one of regularization methods of deep neural network [6], is used to cope with the over-fitting to the small labeled training data. The dropout improves the generalization performance by regularizing the tendency of the co-adaptation of inside nodes that causes the over-fitting via deleting inside nodes randomly [7]. By combining pre-training with unlabeled data and dropout regularization, we can obtain more generalized and accurate neural networks.

In spite of improving training procedure and the performance of classifiers, error occurrence is inevitable and this triggers the error amplification. So, the correction of self-labeled data are necessary to avoid error amplification in the following classifiers. In common self-training schemes, once an example is self-labeled, it is persistently included in the labeled training dataset and used for next classifier construction. This makes a classifier learn improper classification boundaries and generate more misclassified examples. Therefore, we introduce ‘error forgetting’ consists of ‘weight reset’ that ‘example re-evaluation’ that remove examples with low confidence.

This paper is organized as follows. In Section II the related works of self-training are described. Section III introduces the proposed method with our idea. Experimental results are presented in Section IV. Finally, Section V concludes the paper.

2. Related Work

There are some related works of self-training that use various types of training methods. Because the training methods are not designed for self-training originally, there are some problems in applying the training method to the self-training. This section describes how they performed training classifier and what the problems about each approach are.

Tanha et al. [2] proposed a decision tree based self-training approach. This approach uses a label proportion of training labeled examples included in corresponding leaf node as confidence measure of classified example. For example, if 80% of the training labeled examples are 'A' class examples and the other 20% are 'B' class examples in a node, the classification confidence of the examples in the node is 80%. If the number of labeled samples in the nodes is large enough, the confidence is reliable. However, because self-training is usually applied to condition that the number of labeled samples is small and the number of labeled samples corresponding to each leaf nodes is extremely smaller, a decision tree based self-training approach has problem that the confidence is less reliable.

In self-training SVM proposed by Li et al. [3], the confidence measure of self-labeled examples is the distance from the classification boundary. If self-labeled examples are closed to the classification boundary, it is probable that the examples are misclassified. In the case that self-labeled examples are far from classification boundary, it can be considered that misclassification is less probable. However, it is less probable that the selected examples revise classification boundary, because whereas support vectors that determine the classification boundary are closest examples to the classification boundary, the selected examples are far from the classification boundary. By this, examples similar to initial small labeled dataset are only selected and learned consistently, and the classifier is hard to generalize.

Frinken and Bunke [4] used a neural network as base classifier of self-training because of advantages of neural network including requiring less formal statistical knowledge, ability to detect complex nonlinear relationships between input variables, and the availability of incremental training scheme. In this approach, the softmax output of input values are regarded as class probability of each example and be used as estimation of classification confidence. However, because the training of neural network is reducing the difference between the output value and the target value, the selected examples that show little difference between the output value and the target value lead to little revision of classification boundary. In other words, the selected self-labeled examples are not informative for self-training.

Self-training EM algorithm of Nigam et al. [5] used likelihood value of the examples as label confidence. In this approach, a maximum likelihood estimator that is trained using labeled examples calculates the expectation of unlabeled examples about each label, and the labels that have a largest expectation value are assigned to the unlabeled examples. There

are various maximum likelihood estimators corresponding to various data distribution. For getting correct the likelihood of the training examples, a maximum likelihood estimator corresponding to training data should be selected. However, selecting proper maximum likelihood estimator is not easy with small labeled data and depends on expert knowledge and primary data.

3. Proposed Method

In self-training, the error amplification is a major problem. If the current base classifier mislabels some examples, the mislabeled examples may provide inaccurate information to base classifier in the next phase. Then the next base classifier may learn the inaccurate information and generate more mislabeled examples. By this vicious circle, the error of the base classifier is reinforced. In order to reduce the error amplification, the base classifier needs ability to learn accurate but generalized class boundaries even from small training data. In addition, misclassified examples in the proceeding labeling processes are inevitable, so it needs to have a filtering step to remove errors in the classifier or the training dataset. For achieving those, we propose a deep neural network based approach for self-training by adopting pre-training, dropout and error forgetting.

In order to create accurate but generalized classifiers, we use a large amount of unlabeled training data. We adopt a deep belief network (DBN). A DBN is composed of multiple layers of restricted Boltzmann machines, which are a type of probabilistic artificial neural network that can learn a probability distribution over its input set [7–9]. The pre-training guides learning of the randomly initialized neural network to basins of attraction of optima that support better generalization. After pre-training, a supervised learning is performed to make DBNs learn more accurate class boundaries. As a result, with a supervised learning following the pre-training with unlabeled data, reliable self-labeled examples can be generated.

Even though the pre-training is performed before a supervised learning, there is still a challenge that the base classifier can be excessively fitted to the small labeled data. The small labeled data may have partial label information compared to the whole data. The excessively fitted classifier may correctly classify examples similar to small labeled data only and incorrectly classify novel examples not similar to small labeled data. So, the classifier is hard to be generalized and earn enough classification performance even if the pre-training is applied. For alleviating the over-fitting, we adopt another approach, dropout,

to improve the generalization performance. Dropout is a regularization technique for improving the generalization error of large deep neural network [6]. The idea of the dropout is dropping hidden nodes and their connections (weights) randomly in training process by only keeping a neuron active with some probability and setting to zero otherwise. Intuitively, the connections of dropped nodes are not trained temporarily. This regularizes biasing overall weights towards the small datasets and makes the resulting neural network more robust to overfitting. Consequently, we can improve the generalization performance of the self-training using initial small labeled dataset.

In addition to improving the classification performance of classifier, we introduce an ‘error forgetting’ scheme which disposes of connection weights of the classifiers or misclassified examples in the training dataset. We propose two error forgetting schemes: weights reset and example re-evaluation. The weights reset scheme is applied to cope with misconception of the base classifier and the example re-evaluation scheme is applied to remove erroneous examples in the training dataset.

There are two options for training neural network: weight keeping and weight reset. Figure 1 depicts the weight keeping and the weight reset. The weight keeping is using trained weight until current training phase as initial weight of the classifier in next training phase. In Figure 1(a), the classifiers of each phase are trained using the classifier of previous phase successively. Because the neural network requires a large amount of computational cost for learning data, the weight keeping option is needed to train the neural network fast. The weight keeping has a problem that it may aggravate error amplification because misconception generated during previous self-training phase is succeeded to the next self-training phase. If some examples are mislabeled and used to train classifier in next training phase, the error will be persistent in weight keeping option and, it may reinforce error amplification. The weight reset is using pre-trained initial weight as initial weight of the classifier in every self-training phase. In Figure 1(b), the classifiers of each phase are trained using same pre-trained initial weight. The strength of weight reset is that the misconception can be forgotten simply by ridding previous weights. However, it requires a large amount of calculation to learn enough the training data, and iterative performing of this consumes a lot of cost.

We adopt the weight reset to our self-training scheme because the weakness of the weight reset is not problem in our self-training scheme. The reason is that we already have roughly trained weights of the classifier through unsupervised pre-training. As the classifier is pre-trained, it can be converged

Figure 1. The comparison of the weight keeping (a) and the weight reset (b).

Figure 2. Error rate of non-pre-training and pre-training.

fast and doesn't require many training iterations. For confirming this consumption, we performed a toy experiment comparing drop speed of error rate between starting from randomly initialized weights and starting from pre-trained weights with MNIST handwritten dataset [17]. Figure 2 depicts a result of the toy experiment. It is showed that the error rate in case of pre-training is lower than the error rate in case of non-pre-training for same training epoch drop speed of error rate. This means that our assumption is reasonable.

The example re-evaluation scheme targets to remove incorrectly labeled examples already included in self-labeled training examples. Although the classification performance of the classifier is improved, some misclassifying is unavoidable and the classifier may give an incorrect label to examples with a high confidence. Shown Figure 3(a), evaluation targets of self-training are limited to only unlabeled examples. This means that if an example is assigned to a label, the label is persistently

Figure 3. The comparison of common self-training (a) and example re-evaluation (b).

fixed to the example. While self-training process, outputs of classifiers phase vary slightly succeeding phase. This means that the classification confidence of the example can vary while training process. But the common self-training cannot reflect the varying. Because this is probable to trigger the error amplification, the varying of the classification confidence of each example should be labels of the mislabeled examples should be reflected in the classifier. So, we don't limit the evaluation target to only unlabeled examples and re-evaluate the classification confidence the self-labeled examples. Shown Figure 3(b), the evaluation target is both self-labeled examples and unlabeled examples.

Not only each scheme of our approach can lead to improvement of self-training performance, but also the combination of the schemes may lead to more improvement of self-training performance. We will confirm this in following section.

4. Experiment

4.1 Dataset and Experimental Setup

We use three dataset to evaluate the proposed approach: MNIST handwritten digits database [17], smartphone-based recognition of human activities and postural transitions data set (HAR) [18], sensorless drive diagnosis data set (SD) [19]. MNIST handwritten digits database is a standard database for fast-testing handwritten digit recognition. This dataset consists of a training set of 60,000 examples and a test set of 10,000 examples and we perform experiments with this composition. The digits have been size-normalized and centered in a fixed-size image. The dimensionality of each image sample vector is $28 \times 28 = 784$, where each element is binary. The HAR dataset contains the user's actions through the sensors like accelerometer and gyroscope on the smartphone. The HAR dataset consists of a

set of 10,929 examples and 561 features with labels that are 11 different postures of smartphone users. The 90% of the dataset is a training set and the other 10% is a test set and we perform experiments with this composition. The SD dataset consists of features that were extracted from electric current drive conditions: speeds, load moments and load forces. This dataset consists of 19,552 examples with a dimensionality of each instance vector is 48, where each element is real number and each instance is classified into one of 11 classes. We use 90% of the whole dataset as a training set and 10% of the whole dataset as test set.

Except for the MNIST handwritten digits database, the other datasets are in real number. Since the restricted Boltzmann machine is designed for binary value inputs, the real number dataset cannot be processed appropriately. Therefore, in that cases, the pre-training is performed using Gaussian Bernoulli restricted Boltzmann machine (GBRBM) which is one of variations of restricted Boltzmann machine for processing real values [20]. The deep neural network used for experiments consists of three layers and each layer has 500 nodes. The dropout rates are empirically set. For MNIST dataset, they are to 0.5 for all the three layers and, for HAR and SD datasets, they are set to 0.5, 0.15, and 0.2 for each layer.

4.2 Experimental Result

In this sub-section, some experiments are performed by combining training options introduced in Section III: pre-training, dropout, two error forgetting schemes and various combinations. We experiment varying the ratio of initial labeled data in whole training data: 10%, 2%, and 1%. Each combination is represented with following alphabets:

- **P**: Deep neural network with pre-training
- **S**: Self-training
- **D**: Dropout
- **W**: Weight reset scheme
- **E**: Example re-evaluation scheme

Tables 1- 3 show the experiment results about experimental data: MNIST handwritten digits database, HAR dataset, SD dataset. The tables contain the error rate and the rank of the error rate in each combination of schemes and the ratio of initial labeled data in whole training data.

First of all, we compare the performance of self-training with only pre-training (P), the case of self-training with pre-training

(PS), the case of dropout with pre-training (PD) and the case of a combination of self-training and dropout with pre-training (PSD). In many cases, the PS shows low performance (whole average rank: 6.3) than the P (whole average rank: 6.1) though PS consumes more computational cost than P. This is caused by the error amplification that badly affects to classification performance. In contrary, the PD shows better performance (whole average rank: 4.3) than the P or the PS relatively. This shows that the dropout can improve classification performance as generally known. The PSD shows more improved classification performance (whole average rank: 3.3) than individually using the schemes (the PS and the PD). This means that the classifier regularized by dropout makes correct self-labeled examples containing more information than the thing of previous labeled examples and the classifier in the next phase can learn more information from the self-labeled examples. By the repeating this, the classification performance of the self-training is improved.

We also compare the combinations of error forgetting schemes (W, E) with classifier improvement schemes (P, S, D). Generally, a combination of self-training, dropout and weight reset with pre-training (PSDW, whole average rank: 2.3) or a combination of self-training, dropout and example re-evaluation with pre-training (PSDE, whole average rank: 2.4) shows improved performance than self-training and dropout with pre-training (PSD, whole average rank: 3.3). However, the case of PSDWE (whole average rank: 3.0) shows degraded the classification performance than the PSDW or the PSDE. A reason of this degrading is assumed to that the combination of the error forgetting schemes causes over-regularization and the over-regularization excessively discards the delicate information about the classification boundary. In other words, because the error forgetting suppresses the classification boundary to be complex and the example re-evaluation is also fitted it, the self-labeled examples represent only simple classification boundary although real classification boundary has some complex part. For dealing with it, more research will be needed in future work about relations of the re-evaluation and the weight reset.

As a result, in this experiment, the PSDW and the PSDE show the best performance on the most labeled data ratio and dataset. When we compared the performance among P, PS, PD, and PSD, simple adoption of self-training decreases performance but when self-training and dropout are applied together, self-training significantly improves performance. In addition, when we compared PSD, PSDW, and PSDE, error forgetting schemes increase performance because it could prevent the error

Table 1. Error (%) rate of the models on MNIST dataset

Labeled data ratio	10		2		1		Average rank
	Error (%)	Rank	Error (%)	Rank	Error (%)	Rank	
P	3.91	6	6.99	7	8.54	7	6.7
PS	4.20	7	5.82	5	6.94	4	5.3
PD	3.06	5	5.80	4	7.82	6	5
PSD	2.85	3	5.69	3	5.59	1	2.3
PSDW	2.77	2	5.20	2	5.76	2	2.0
PSDE	2.74	1	4.77	1	7.76	5	2.3
PSDWE	2.99	4	6.08	6	6.20	3	4.3

Table 2. Error (%) rate of the models on HAR dataset

Labeled data ratio	10		2		1		Average rank
	Error (%)	Rank	Error (%)	Rank	Error (%)	Rank	
P	5.87	6	14.86	7	18.44	4	5.7
PS	6.66	7	13.79	6	21.46	7	6.7
PD	4.86	5	13.21	4	16.05	2	3.7
PSD	4.35	2	12.87	3	19.90	6	3.7
PSDW	4.35	2	13.42	5	19.25	5	4.0
PSDE	4.07	1	12.77	2	16.11	3	2.0
PSDWE	4.44	4	11.38	1	15.00	1	2.0

Table 3. Error (%) rate of the models on SD dataset

Labeled data ratio	10		2		1		Average rank
	Error (%)	Rank	Error (%)	Rank	Error (%)	Rank	
P	5.48	6	16.07	6	28.92	6	6.0
PS	5.56	7	16.44	7	33.86	7	7.0
PD	5.12	4	15.74	5	24.05	4	4.3
PSD	5.34	5	12.15	2	27.58	5	4.0
PSDW	3.48	1	10.92	1	19.38	1	1.0
PSDE	3.89	2	14.30	4	21.89	3	3.0
PSDWE	4.30	3	12.46	3	20.05	2	2.7

amplification.

5. Conclusion

In this paper, we propose the self-training scheme adopting unsupervised pre-training based on restricted Boltzmann machine learning methods, dropout. The target of our approach is alleviating error amplification that is major problem of self-training.

By applying combinations of these schemes, the classification performance of self-training is improved in various datasets and it is showed and confirmed in experimental result. However, the combination of the error forgetting and example re-evaluation shows the performance degradation in our experiments. We assume that the reason of the performance degradation is the over-regularization by the combination of the error forgetting and example re-evaluation and we will perform more research

about the over-regularization in future work.

Conflict of Interest

No potential conflict of interest relevant to this article was reported.

Acknowledgments

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (No. NRF-2016R1A2B4015820).

References

- [1] X. Zhu and A. B. Andrew, "Introduction to semisupervised learning," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 3, no. 1, pp. 1-130, 2009. <http://dx.doi.org/10.2200/S00196ED1V01Y200906AIM006>
- [2] J. Tanha, M. van Someren, and H. Afsarmanesh, "Semi-supervised self-training for decision tree classifiers," *International Journal of Machine Learning and Cybernetics*, vol. 8, no. 1, pp. 355-370, 2017.
- [3] Y. Li, C. Guan, H. Li, and Z. Chin, "A self-training semi-supervised SVM algorithm and its application in an EEG-based brain computer interface speller system," *Pattern Recognition Letters*, vol. 29, no. 9, pp. 1285-1294, 2008. <http://dx.doi.org/10.1016/j.patrec.2008.01.030>
- [4] V. Frinken and H. Bunke, "Evaluating retraining rules for semi-supervised learning in neural network based cursive word recognition," in *Proceeding of the 10th International Conference on Document Analysis and Recognition*, Barcelona, Spain, 2009, pp. 31-35. <http://dx.doi.org/10.1109/ICDAR.2009.18>
- [5] K. Nigam, A. McCallum, and T. M. Mitchell, "Semi-supervised text classification using EM," in *Semi-Supervised Learning*. Cambridge, MA: MIT Press, 2016, pp. 33-56.
- [6] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle, "Greedy layer-wise training of deep networks," *Advances in Neural Information Processing Systems*, vol. 19, pp. 153-160, 2006.
- [7] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," Available <https://arxiv.org/1207.0580v1>
- [8] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359-366, 1989. [http://dx.doi.org/10.1016/0893-6080\(89\)90020-8](http://dx.doi.org/10.1016/0893-6080(89)90020-8)
- [9] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436-444, 2015. <http://dx.doi.org/10.1038/nature14539>
- [10] A. Coates, H. Lee, and, A. Y. Ng, "An analysis of single-layer networks in unsupervised feature learning," in *Proceeding of the 14th International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, FL, 2011, pp. 215-223.
- [11] P. Smolensky, "Information processing in dynamical systems: foundations of harmony theory," in *Parallel Distributed Processing Volume 1*. Cambridge, MA: MIT Press, 1986, pp. 194-281.
- [12] Y. Freund and D. Haussler, "Unsupervised learning of distributions on binary vectors using two layer networks," in *Proceedings of the 4th International Conference on Neural Information Processing Systems*, Denver, CO, 1992, pp. 912-919.
- [13] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, no. 6, pp. 721-741, 1984. <http://dx.doi.org/10.1109/TPAMI.1984.4767596>
- [14] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Computation*, vol. 14, no. 8, pp. 1771-1800, 2002. <http://dx.doi.org/10.1162/089976602760128018>
- [15] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proceedings of Neural Information Processing Systems*, Lake Tahoe, CA, 2012, pp. 1097-1105.
- [16] F. Li, L. Tran, K. H. Thung, S. Ji, D. Shen, and J. Li, "Robust deep learning for improved classification

of AD/MCI patients,” in *Machine Learning in Medical Imaging*. Cham: Springer, 2014, pp. 240-247. http://dx.doi.org/10.1007/978-3-319-10581-9_30

- [17] J. Li, X. Wang, and B. Xu, “Understanding the dropout strategy and analyzing its effectiveness on LVCSR,” in *Proceeding of 2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, Canada, 2013, pp. 7614-7618. <http://dx.doi.org/10.1109/ICASSP.2013.6639144>
- [18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278-2324, 1998. <http://dx.doi.org/10.1109/5.726791>
- [19] D. Anguita, A. Ghio, L. Oneto, X. Parra, and J. L. Reyes-Ortiz, “A public domain dataset for human activity recognition using smartphones,” in *Proceedings of 2013 European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning*, Bruges, Belgium, 2013, pp. 437-442.
- [20] C. Bayer, O. Enge-Rosenblatt, M. Bator, and U. Monks, “Sensorless drive diagnosis using automated feature extraction, significance ranking and reduction,” in *Proceeding of 2013 IEEE 18th Conference on Emerging Technologies & Factory Automation*, Cagliari, Italy, 2013, pp. 1-4. <http://dx.doi.org/10.1109/ETFA.2013.6648126>
- [21] D. H. Ackley, G. E. Hinton, and T. J. Sejnowski, “A learning algorithm for Boltzmann machines,” *Cognitive Science*, vol. 9, no. 1, pp. 147-169, 1985. [http://dx.doi.org/10.1016/S0364-0213\(85\)80012-4](http://dx.doi.org/10.1016/S0364-0213(85)80012-4)

Hye-Woo Lee received his B.S. and M.S. in computer engineering from Sungkyunkwan University, Suwon, Korea, in 2014 and 2016, respectively. He is currently research engineer at KT R&D Center, KT. His current research interests are semi-supervised learning and machine learning.

E-mail: lee.hyewoo@kt.com

Noo-ri Kim received his B.S. in computer engineering from Sungkyunkwan University, Suwon, Korea, in 2013. He is currently pursuing his M.S. and Ph.D. in computer engineering at Sungkyunkwan University. His research interests include group recommender system and machine learning.

E-mail: pd99j@skku.edu

Jee-Hyong Lee received his B.S., M.S., and Ph.D. in computer science from the Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1993, 1995, and 1999, respectively. From 2000 to 2002, he was an international fellow at SRI International, USA. He joined Sungkyunkwan University, Suwon, Korea, as a faculty member in 2002. His research interests include fuzzy theory and application, intelligent systems, and machine learning.

E-mail: john@skku.edu