IPSJ Transactions on Mathematical Modeling and Its Applications Vol.10 No.2 25–32 (July 2017)

Regular Paper

Analysis of Conventional Dropout and its Application to Group Dropout

Kazuyuki Hara[1,a)] Daisuke Saitoh[2] Satoshi Suzuki[3] Takumi Kondou[2] Hayaru Shouno[3]

Received: November 4, 2016, Revised: January 1, 2017,
Accepted: February 7, 2017

Abstract: Deep learning is a state-of-the-art learning method that is used in fields such as visual object recognition and speech recognition. It uses very deep layers and a huge number of units and connections, so overfitting is a serious problem. The dropout method is used to address this problem. Dropout is a regularizer that neglects randomly selected inputs and hidden units during the learning process with probability q; after learning, the neglected inputs and hidden units are combined with the learned network to express the final output. Wager et al. pointed out that conventional dropout is an adaptive L2 regularizer, so we compared the learning behavior of conventional dropout with that of stochastic gradient descent with the L2 regularizer. We found that combining the neglected hidden units with the learned network can be regarded as ensemble learning, so we analyzed, on the basis of on-line learning, conventional dropout learning from the viewpoint of ensemble learning. Next we compared conventional dropout and ensemble learning from two additional viewpoints and confirmed that conventional dropout can be regarded as ensemble learning that divides a student network into two sub-networks. On the basis of this finding, we developed a novel dropout method that divides the network into more than two sub-networks. Computer simulation demonstrated that this method enhances the benefit of ensemble learning.

Keywords: dropout, over-fitting, ensemble learning, online learning, soft-committee machine

1. Introduction

Deep learning [1], [2] is attracting much attention in visual object recognition, speech recognition, object detection, and many other fields. It provides automatic feature extraction and can achieve outstanding performance [3].

Deep learning uses a very deep layered network and a huge amount of training data, so overfitting is a serious problem. To avoid overfitting, the conventional dropout method [3] is used for regularization. Conventional dropout consists of two processes. During learning, randomly selected hidden units are neglected with probability q, thereby reducing the network size; therefore, this relaxs overfitting. During testing, the learned and unlearned sub-networks are summed up and multiplied by $p = 1 - q$ to calculate the network output. Hinton observed that dropout seems like a type of ensemble learning. Wager et al. pointed out that dropout is an adaptive L2 regularizer.

Baldi et al. showed that the result of conventional dropout is approximated by the normalized weighted geometric mean [4]. Since the weighted geometric mean is related to ensemble learning, conventional dropout is also related to ensemble learning. Ensemble learning improves the performance of a single network by using the average for many networks. Bagging and the Ada-boost algorithm are well known as a type of ensemble learning [5]. We theoretically analyzed ensemble learning using linear or non-linear perceptrons [6], [7].

In this paper, we first present our analysis of conventional dropout as a regularizer and then as a type of ensemble learning. On-line learning [8], [9] is used to learn a network. To estimate the regularization performance of conventional dropout, we compared the residual error of conventional dropout with that of the stochastic gradient descent (SGD) algorithm with L2 regularization [10]. Next, we compared the learnability of conventional dropout with that of ensemble learning using the same network structure [10]. The results revealed that conventional dropout can be regarded as ensemble learning of learned network and unlearned network. After that, we present a novel dropout method called "group dropout" [11], [12]. The proposed method divides the hidden units in a student network into several sub-networks, and each sub-network learns from the teacher independently and simultaneously. After the learning, the group outputs are averaged to obtain the student output. The proposed method thereby enhances ensemble learning compared with that of conventional dropout. Finally, we present the results of computer simulation, which demonstrate the validity of the proposed method.

2. Formulation

2.1 Model

Here, networks are learned by on-line learning. We use a teacher-student formulation and assume the existence of a teacher

[1] College of Industrial Technology, Nihon University, Narashino, Chiba 275–8575, Japan
[2] Graduate School of Industrial Technology, Nihon University, Narashino, Chiba 275–8575, Japan
[3] Graduate School of Informatics and Engineering, The University of Electro-Communications, Chofu, Tokyo 182–8585, Japan
[a)] hara.kazuyuki@nihon-u.ac.jp

© 2017 Information Processing Society of Japan

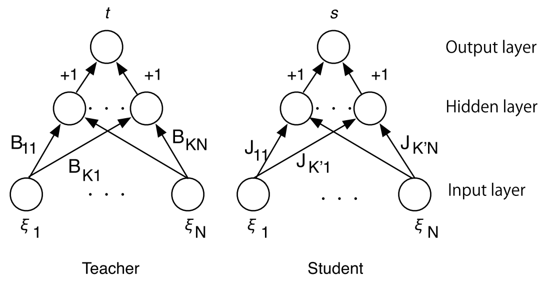

Fig. 1 Network structures of teacher and student.

that produces the output desired for training data. By introducing a teacher, we can directly measure the similarity of the student weight vector to that of the teacher. First, we formulate a teacher network (referred to as the "teacher") and a student network (referred to as the "student") and then introduce the SGD algorithm.

The teacher and student are a soft committee machine with N input units, several hidden units, and an output, as shown in **Fig. 1**. The teacher consists of K hidden units, and the student consists of K' hidden units. Each hidden unit is a perceptron. The kth hidden weight vector of the teacher is $\boldsymbol{B}_k = (B_{k1}, \dots, B_{kN})$, and the k'th hidden weight vector of the student is $\boldsymbol{J}_{k'}^{(m)} = (J_{k'1}^{(m)}, \dots, J_{k'N}^{(m)})$, where m denotes the number of learning iterations. In the soft committee machine, all hidden-to-output weights are fixed to +1 [9]. This network calculates the majority vote of the hidden outputs.

We assume that both the teacher and the student receive N-dimensional training data $\boldsymbol{\xi}^{(m)} = (\xi_1^{m}, \dots, \xi_N^{(m)})$ and that the teacher outputs $t^{(m)}$ and the student outputs $s^{(m)}$ are

$$t^{(m)} = \sum_{k=1}^{K} t_k^{(m)} = \sum_{k=1}^{K} g(d_k^{(m)}), \tag{1}$$

$$s^{(m)} = \sum_{k'=1}^{K'} s_{k'}^{(m)} = \sum_{k'=1}^{K'} g(y_{k'}^{(m)}), \tag{2}$$

where $g(\cdot)$ is the activation function of a hidden unit, $d_k^{(m)}$ is the inner potential of the kth hidden unit of the teacher, and $y_{k'}^{(m)}$ is the inner potential of the k'th hidden unit of the student:

$$d_k^{(m)} = \sum_{i=1}^{N} B_{ki}\xi_i^{(m)}, \tag{3}$$

$$y_{k'}^{(m)} = \sum_{i=1}^{N} J_{k'i}^{(m)}\xi_i^{(m)}. \tag{4}$$

We assume that the ith elements $\xi_i^{(m)}$ of the independently drawn training data $\boldsymbol{\xi}^{(m)}$ are uncorrelated random variables with zero mean and unit variance; that is, the ith element of the training data is drawn from probability distribution $\mathrm{P}(\xi_i)$. The thermodynamic limit of $N \to \infty$ is also assumed. The statistics of training data $\boldsymbol{\xi}^{(m)}$ at the thermodynamic limit of $N \to \infty$ are

$$\left\langle \xi_i^{(m)} \right\rangle = 0, \left\langle (\xi_i^{(m)})^2 \right\rangle \equiv \sigma_\xi^2 = 1, \left\langle \|\boldsymbol{\xi}^{(m)}\| \right\rangle = \sqrt{N}, \tag{5}$$

where $\langle\cdot\rangle$ denotes the mean, and $\|\cdot\|$ denotes the norm of a vector.

For each element B_{ki}, $k = 1 \sim K$ is drawn from a probability distribution with zero mean and $1/N$ variance. With the assumption of the thermodynamic limit, the statistics of the teacher weight vector are

$$\langle B_{ki} \rangle = 0, \left\langle (B_{ki})^2 \right\rangle \equiv \sigma_B^2 = \frac{1}{N}, \langle \|\boldsymbol{B_k}\| \rangle = 1.$$

This means that any combination of $\boldsymbol{B}_l \cdot \boldsymbol{B}_{l'} = 0$. The distribution of inner potential $d^{(m)}$ follows a Gaussian distribution with zero mean and unit variance at the thermodynamic limit.

For the sake of analysis, we assume that for each element $J_{k'i}^{(0)}$, $k' = 1 \sim K'$ which is the initial value of the student weight vector $\boldsymbol{J}_{k'}^{(0)}$, is drawn from a probability distribution with zero mean and $1/N$ variance. At the thermodynamic limit, the statistics of the initial value of the student weight vector are

$$\left\langle J_{k'i}^{(0)} \right\rangle = 0, \left\langle (J_{k'i}^{(0)})^2 \right\rangle \equiv \sigma_J^2 = \frac{1}{N}, \left\langle \|\boldsymbol{J}_{k'}^{(0)}\| \right\rangle = 1.$$

This means that any combination of $\boldsymbol{J}_l^{(0)} \cdot \boldsymbol{J}_{l'}^{(0)} = 0$. The activation function of the hidden units of the student $g(\cdot)$ is the same as that of the teacher. The statistics of the student weight vector at the mth iteration are

$$\left\langle J_{k'i}^{(m)} \right\rangle = 0, \left\langle (J_{k'i}^{(m)})^2 \right\rangle = \frac{(Q_{k'k'}^{(m)})^2}{N}, \left\langle \|\boldsymbol{J}_{k'}^{(m)}\| \right\rangle = Q_{k'k'}^{(m)}.$$

Here,

$$(Q_{k'k'}^{(m)})^2 = \boldsymbol{J}_{k'}^{(m)} \cdot \boldsymbol{J}_{k'}^{(m)}.$$

The distribution of the inner potential $y_{k'}^{(m)}$ follows a Gaussian distribution with zero mean and $(Q_{k'k'}^{(m)})^2$ variance in the thermodynamic limit.

2.2 On-line Learning on Soft Committee Machine

Next, we introduce the SGD algorithm for the soft committee machine. For the possible training data $\{\boldsymbol{\xi}\}$, we want to train the student to produce the desired outputs, $t = s$. The generalization error is defined as the squared error ε averaged over possible training data:

$$\begin{aligned}\varepsilon_g^{(m)} &= \left\langle \varepsilon^{(m)} \right\rangle = \frac{1}{2}\left\langle (t^{(m)} - s^{(m)})^2 \right\rangle \\ &= \frac{1}{2}\left\langle \left(\sum_{k=1}^{K} g(d_k^{(m)}) - \sum_{k'=1}^{K'} g(y_{k'}^{(m)}) \right)^2 \right\rangle. \end{aligned} \tag{6}$$

At each learning step m, a new uncorrelated training data instance, $\boldsymbol{\xi}^{(m)}$, is presented, and the current hidden weight vector of the student $\boldsymbol{J}_{k'}^{(m)}$ is updated using

$$\begin{aligned}\boldsymbol{J}_{k'}^{(m+1)} &= \boldsymbol{J}_{k'}^{(m)} + \frac{\eta}{N}\left(\sum_{l=1}^{K} g(d_l^{(m)}) - \sum_{l'=1}^{K'} g(y_{l'}^{(m)}) \right) \\ &\quad \times g'(y_{k'}^{(m)})\boldsymbol{\xi}^{(m)}, \end{aligned} \tag{7}$$

where η is the learning step size and $g'(x)$ is the derivative of the activation function of the hidden unit $g(x)$.

The on-line learning framework is based on the assumption of an infinite size training data set with independent generation, so that overfitting is not theoretically considered. However, to address overfitting in the on-line learning framework, we assume that the size of the training data set is limited to 10× N data instances are generated. We also assume that the training patterns in the generated data are reused in the training phase. This assumption holds hereafter.

3. Analysis of Conventional Dropout

Conventional dropout [3] is used in deep learning to prevent overfitting. A small amount of training data compared with the network size may cause overfitting [13], and overfitting can cause the training error to differ from the test error. Hereafter, we denote "training errors" and "test errors" as errors for the training and test data respectively. We assume that the test errors are independent of the training data. The conventional dropout learning equation for the soft committee machine can be written as

$$\boldsymbol{J}_{k'}^{(m+1)} = \boldsymbol{J}_{k'}^{(m)} + \frac{\eta}{N}\left(\sum_{l=1}^{K} g(d_l^{(m)}) - \sum_{l' \in D^{(m)}}^{pK'} g(y_{l'}^{(m)})\right) \times g'(y_{k'}^{(m)})\boldsymbol{\xi}^{(m)}, \tag{8}$$

where $D^{(m)}$ includes a number of hidden units that are randomly selected with probability p from all hidden units at the mth iteration. Subscript k' of the student weight vector $\boldsymbol{J}$ is included in $D^{(m)}$. Note that the second term in the brackets on the right hand side of Eq. (8) is a soft committee machine composed of selected hidden units. The hidden units in $D^{(m)}$ are subject to learning, so the size of the student decreases, and a smaller student may be immune from overfitting. This effect is the conventional dropout opportunity. After the learning, the student's output, $s^{(m)}$, is calculated as the sum of the learned and unlearned hidden outputs multiplied by p.

$$s^{(m)} = p * \left\{\sum_{l' \in D^{(m)}}^{pK'} g(y_{l'}^{(m)}) + \sum_{l' \notin D^{(m)}}^{qK'} g(y_{l'}^{(m-1)})\right\} \tag{9}$$

This equation is regarded as the ensemble of a learned soft committee machine (first term on right hand side) and that of an unlearned soft committee machine (second term on right) when the probability is $p = 0.5$. However, in conventional deep learning, the set of hidden units in $D^{(m)}$ is changed at every iteration, and the same set of hidden units is used in the ensemble learning. Therefore, conventional dropout is regarded as ensemble learning using a different set of hidden units at every iteration. Therefore, we refer to conventional dropout as "random dropout" in this paper.

Figure 2 shows the results of the SGD algorithm without regularization and those of random dropout. The soft committee machine was used for both the teacher and student. A sigmoid-like function, $\mathrm{erf}(x/\sqrt{2})$, was used as the activation function, $g(x)$. We generated $10 \times N$ training data instances and N test data instances. The number of inputs was $N = 1{,}000$. The teacher had two hidden units, and the student had 100 hidden units. The training data and their target were generated as described in Section 2. Learning step size η was set to 0.01. The horizontal axis is time, $\alpha = m/N$, where m is the iteration number, and N is the number of input units. The vertical axis shows the normalized mean squared error (MSE) for input scale N.

Figure 2 (a) shows the learning curve of the SGD algorithm without regularization. In this setting, overfitting will occur. Figure 2 (b) shows the learning curve of the SGD algorithm with dropout. The learning error was less than the test error; however, the difference between the training error and the test error was not as substantial as that of the SGD algorithm (Fig. 2 (a)). Therefore, these results show that random dropout prevents overfitting.

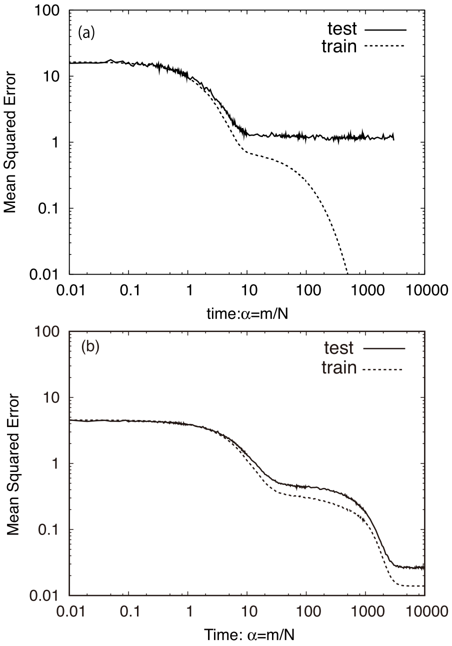

Fig. 2 Effect of dropout: (a) learning curve of SGD algorithm; (b) learning curve of dropout learning.

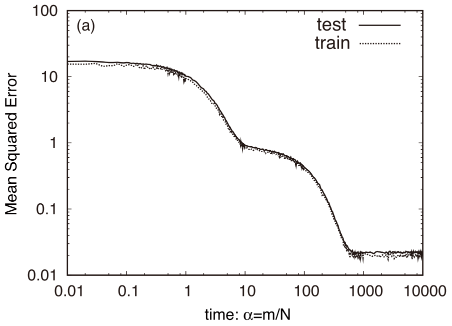

Fig. 3 Learning curve of SGD with L2.

3.1 Comparison between Dropout and SGD Algorithm with L2 Regularization

As mentioned above, Wager et al. pointed out that random dropout is an adaptive L2 regularizer [14]. Thus, in this subsection, we present a comparison of dropout and the SGD algorithm with L2 regularization ("SGD with L2"), which is represented here as

$$\boldsymbol{J}_{k'}^{(m+1)} = \boldsymbol{J}_{k'}^{(m)} + \frac{\eta}{N}\left(\sum_{l=1}^{K} g(d_l^{(m)}) - \sum_{l'=1}^{K'} g(y_{l'}^{(m)})\right) \times g'(y_{k'}^{(m)})\boldsymbol{\xi}^{(m)} - \gamma \boldsymbol{J}_{k'}^{(m)}, \tag{10}$$

γ is the coefficient of the L2 penalty. As shown, L2 penalty decreases $\|\boldsymbol{J}_{k'}^{(m)}\|$.

Figure 2 (b) and **Fig. 3** show the learning results of dropout and of SGD with L2. We used soft committee machines that included 100 hidden units. Activation function $g(x)$ is a sigmoid-like function $\mathrm{erf}(x/\sqrt{2})$. For random dropout, we set $p = 0.5$. For SGD with L2, we selected $\gamma = 10^{-6}$ as the optimum coefficient. The learning step size was set to $\eta = 0.01$. We prepared $10 \times N$ patterns for the training data and N patterns for the test data. Training

© 2017 Information Processing Society of Japan

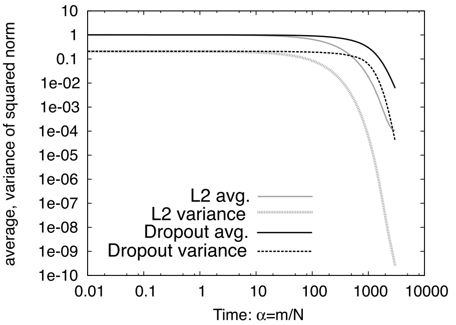

Fig. 4 Squared norms of SGD with L2 and of random dropout.

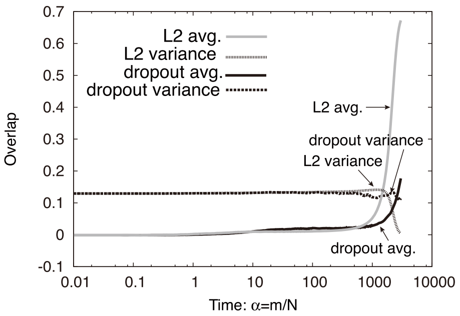

Fig. 5 Overlap of SGD with L2 regularization and that of random dropout.

data were frequently reused in the training. Results were obtained using an average of ten trials.

Comparison of Fig. 2 (b) and Fig. 3 reveals that, under our assumptions, the residual error of random dropout was almost the same as that of SGD with L2. These results suggest that the regularization performance of random dropout is close to that of SGD with L2; however, their strategies are very different. For SGD with L2, we must choose tuning parameter γ through trials whereas random dropout learning has no tuning parameter.

Figure 4 shows the time course of the average and variance of the squared norm of the student weight vector $||\boldsymbol{J}_k||^2$ that were used in Fig. 2 (b) and Fig. 3. The vertical axes are the squared norm of the student weight. The solid lines are the averages of $||\boldsymbol{J}_{k'}||^2$ ("L2 avg." and "Dropout avg."), and the broken lines are the averages of the variances ("L2 variance" and "Dropout variance"). The $||\boldsymbol{J}_{k'}||^2$ decreased as learning proceeded for both SGD with L2 and random dropout. As shown in Eq. (10), L2 penalty decreases $||\boldsymbol{J}_{k'}||$. Therefore, regularization is effective for both methods. The average of $||\boldsymbol{J}_{k'}||^2$ for SGD with L2 was smaller than for random dropout, so regularization is more effective with SGD with L2 than with random dropout. However, the variance of $||\boldsymbol{J}_{k'}||^2$ for random dropout was higher than that for SGD with L2. This means that the diversity of hidden unit outputs when using random dropout is maintained. This may be an advantage for ensemble learning.

Next, we present our investigation of the time course of the overlap of $R_{kk'}$. $R_{kk'}$:

$$R_{kk'} = \boldsymbol{B}_k \cdot \boldsymbol{J}_{k'}. \qquad (11)$$

By measuring $R_{kk'}$ at each time α, we can understand the learning dynamics of SGD with L2 and that of random dropout. **Figure 5** shows the results obtained using $\boldsymbol{B}_1$ and $\boldsymbol{J}_{k'}$, where $k' = 1 \sim 100$. The average of $R_{1k'}$ for SGD with L2 at $t = 3{,}000$ was about $\overline{R} = 0.67$, and that of random dropout was $\overline{R} = 0.17$. The variance of $R_{1k'}$ for SGD with L2 at $t = 3{,}000$ was about $\sigma_R^2 = 4.5 \times 10^{-3}$, and that of random dropout was about $\sigma_R^2 = 1.1 \times 10^{-1}$. These results indicate that SGD with L2 may move all the student weight vectors toward $\boldsymbol{B}_1 + \boldsymbol{B}_2$ because $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ are orthogonal with each other, and $\boldsymbol{B}_1 + \boldsymbol{B}_2$ is located at their mid-point. The overlap between $\boldsymbol{B}_1$ and $\boldsymbol{B}_1 + \boldsymbol{B}_2$ is $\cos(\frac{\pi}{4})$, i.e., ~ 0.71. This is close to $\overline{R} = 0.67$. For random dropout, $\overline{R} = 0.17$ and $\sigma_R^2 = 1.1 \times 10^{-1}$. The effect of ensemble learning using students with small overlap is higher than that with a large overlap [6]. The σ_R^2 of random dropout is high, so the diversity of hidden unit outputs when using random dropout is maintained. These results show that the regularization performance of SGD with L2 and that of random dropout are almost the same; however, their strategies are very different. Random dropout is more suitable for ensemble learning. Therefore, we investigated random dropout as a type of ensemble learning, as described in next subsection.

3.2 Ensemble Learning

Baldi et al. showed that the average properties of the result of random dropout are characterized by approximation of expectations by using the normalized weighted geometric mean [4]. The normalized geometric mean is strongly related to ensemble learning. The geometric mean can applied to a positive value, so Baldi used a sigmoid function, $g(x) = 1/(1 + \exp(-x))$, as the activation function. We used $g(x) = \mathrm{erf}(x/\sqrt{2})$, so we cannot use the geometric mean. However, since the geometric mean is related to the numerical mean, we analyzed ensemble learning by using the numerical mean.

Ensemble learning is performed by using many learners (referred to as "students") to achieve better performance [6]. In ensemble learning, each student learns from the teacher independently simultaneously, and student outputs $s_{k'_{en}}$ are averaged to calculate the ensemble output s_{en}. We assume that the teacher and students are soft committee machines. Thus, the ensemble output s_{en} is calculated using

$$s_{en} = \sum_{k'_{en}=1}^{K_{en}} C_{k'_{en}} s_{k'_{en}} = \sum_{k'_{en}=1}^{K_{en}} C_{k'_{en}} \sum_{k'=1}^{K'} g(y_{k'}), \qquad (12)$$

where K' is the number of hidden units in the students, $C_{k'_{en}}$ is a weight for averaging, and K_{en} is the number of students to be averaged. The learning equation of ensemble learning is the same as Eq. (7).

There are three cases for setting the number of hidden units in the students: (1) $K' < K$, (2) $K' = K$, and (3) $K' > K$. The case of $K' < K$ is unlearnable and insufficient because the degree of complexity of the students is less than that of the teacher. The case of $K' = K$ is learnable because the degree of complexity of the students is the same as that of the teacher. The case of $K' > K$ is learnable and redundant because the degree of complexity of the students is higher than that of the teacher [13]. **Figure 6** shows the time course of the MSE for different settings of students. The teacher includes two hidden units ($K = 2$), and the students include $K' = 2, 10, 20, 30$, or 40. The activation function, $g(x)$, is the error function $\mathrm{erf}(x/\sqrt{2})$. The horizontal axis shows the time

© 2017 Information Processing Society of Japan

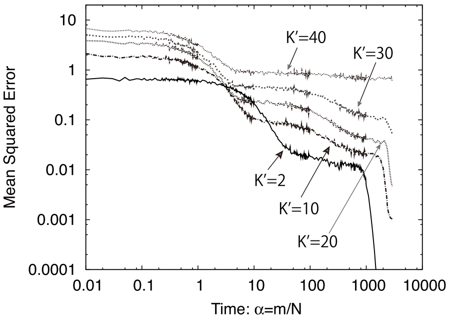

Fig. 6 Effect of different number of hidden units in students ($K = 2$ and $K' = 2, 10, 20, 30$, or 40).

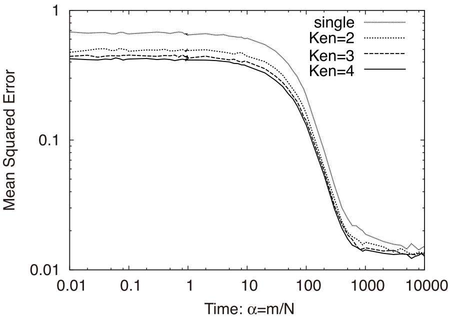

Fig. 7 Effect of ensemble learning.

($\alpha = m/N$), and the vertical axis shows the normalized MSE for input scale N ($N = 1{,}000$). The learning step size was set to $\eta = 0.1$. The MSE decreased the fastest when $K = K'$ ($K' = 2$ in the figure). When $K < K'$, the convergence time increased for larger K'. Therefore, setting $K' = K$ will optimize network performance.

Next, we show the effect of ensemble learning for different numbers of students. **Figure 7** shows the computer simulation results. Each student has the same architecture as the teacher, and each student includes two hidden units, i.e., $K = K' = 2$. The activation function, $g(x)$, is the error function, $\mathrm{erf}(x/\sqrt{2})$. We generated $10 \times N$ training data $\boldsymbol{\xi}^{(m)}$ instances, where $N = 10{,}000$, and they were reused in training. Each of the elements $\xi_i^{(m)}$ of the independently drawn training data $\boldsymbol{\xi}^{(m)}$ were uncorrelated random variables with zero mean and unit variance, as shown in Eq. (5). We also generated N test data. The target for training data instance $\boldsymbol{\xi}$ is the output of the teacher. In the figure, the horizontal axis shows the time, $\alpha = m/N$, where m is the iteration number, and N is the number of input units. The vertical axis shows the normalized MSE for the input scale, N. The MSE was calculated for the test data, which had N independent patterns. In the figure, "single" is the results of using a single student. "Ken=2" is the results of using an ensemble of two students, "Ken=3" is that of an ensemble of three students, and "Ken=4" is that of an ensemble of four students. The MSEs for the test data are plotted. The performance of the ensemble improved when a larger number of students was used. Therefore, the ensemble of four students outperformed the other two ensembles.

Next, we modified the ensemble learning. We assume that student has more hidden units than that of the teacher. We divided the students (with K' hidden units) into K_{en} sub-networks (See **Fig. 8**. Here, $K' = 4$ and $K_{en} = 2$). These sub-networks learned from the teacher independently simultaneously. We calculated ensemble output s_{en} by averaging the outputs of the sub-networks, $s_{k'_{en}}$:

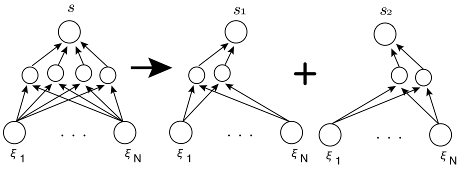

Fig. 8 Network divided into two sub-networks (s_1 and s_2) to apply ensemble learning.

$$s_{en} = \frac{1}{K_{en}} \sum_{k'_{en}=1}^{K_{en}} s_{k'_{en}} = \frac{1}{K_{en}} \sum_{k'_{en}=1}^{K_{en}} \sum_{l'=1}^{K} g(y_{k'_{en}l'}), \tag{13}$$

where $s_{k'_{en}}$ is the output of a sub-network with K hidden units, and $g(y_{k'_{en}l'})$ is the l'th hidden output in the k'_{en}th sub-network. Equation (13) corresponds to Eq. (12) when $C_{k'_{en}} = \frac{1}{K_{en}}$ and $K' = K$.

The next section presents our comparison of random dropout and ensemble learning to clarify the effect of the random selection of hidden units.

3.3 Comparison between Random Dropout and Ensemble Learning

We compared random dropout and ensemble learning from three viewpoints: (1) selecting the hidden units in a sub-network randomly or using the same hidden units, (2) dividing the student into two or more sub-networks that contain some of the hidden units, and (3) averaging the outputs of learned and unlearned sub-networks or averaging only the output of learned sub-networks. Random dropout involves selecting the hidden units in a sub-network randomly, dividing the student into two sub-networks and learning one sub-network, and averaging the output of learned sub-network and that of unlearned sub-network. Ensemble learning involves using the same hidden units in a sub-network throughout the learning, dividing the students into more than two sub-networks, and averaging the output of learned sub-networks. Thus, this subsection concentrates on the effect of selecting the hidden units in a sub-network.

For ensemble learning, we used 2 soft committee machines with 50 hidden units. For random dropout, we used 100 hidden units and set $p = 0.5$; random dropout thus selected 50 hidden units in $D^{(m)}$, with 50 unselected hidden units remaining. Therefore, random dropout and ensemble learning had the same architectures. The number of input units N was 1,000, and the learning step size η was set to 0.01. The activation function, $g(x)$, was a sigmoid-like function, $\mathrm{erf}(x/\sqrt{2})$. The training data and corresponding targets were generated as described in Section 2. We used $10 \times N$ patterns for training and N for testing.

Figure 2 (b) and **Fig. 9** show the results obtained by taking the average of the results of ten trials. The number of training data instances was $10 \times N$, and the number of test data instances was N. To calculate the MSE, N training data instances were randomly selected from $10 \times N$ training data instances. The horizontal axes are time $\alpha = m/N$, and the vertical axes are the MSE calculated for N data instances. Figure 9 shows the time courses of the MSE for the training data and the test data. Two soft com-

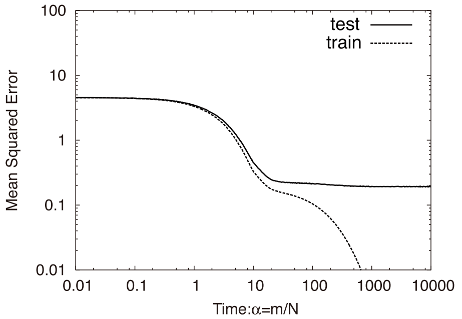

Fig. 9 Learning curve of ensemble learning.

mittee machines with 50 hidden units were used. Figure 9 shows that the error for "test" was larger than that for "train," indicating that overfitting occurred. Figure 2 (b) shows that random dropout had an MSE less than that of ensemble learning. Moreover, the error for "test" and that for "train" were almost the same, so overfitting did not occur with random dropout. Therefore, random dropout outperformed ensemble learning. This happened because it uses randomly selected hidden units and averages the outputs of learned and unlearned sub-networks.

4. Group Dropout

In random dropout, during testing, the output of the learned sub-network and that of unlearned sub-network are summed up and multiply by p to calculate the student output, as shown in Eq. (9). This is regarded as ensemble learning using only two students. In ensemble learning, using more students improves performance. Therefore, the ensemble effect of random dropout can increase by using more than two sub-networks. Therefore, we developed group dropout, which involves dividing the student into more than two sub-networks (see **Fig. 10**) and applying ensemble learning.

As discussed in Section 3.2, the performance of ensemble learning is optimized when the number of teacher hidden units K and that of student K' are equal. Given this result, in group dropout, we divided the hidden units in the student into $K_{gd} = K'/K$ sub-networks before the learning was started. Therefore, the number of hidden units in each sub-network was the same as that in teacher. In group dropout, we assume that each sub-network has the same number of hidden units when dividing the student into sub-networks (see Fig. 10). From the statistical symmetry of the model, our analysis has generality under this assumption.

Each sub-network (s_1, s_2, and s_3 in Fig. 10) learned from the teacher independently and simultaneously. We calculated the ensemble output s_{en} by averaging sub-network outputs s_{gd}:

$$s_{gd} = \frac{1}{K_{gd}} \sum_{k'_{gd}=1}^{K_{gd}} s_{k'_{gd}} = \frac{1}{K_{gd}} \sum_{k'_{gd}=1}^{K_{gd}} \sum_{k'=1}^{K} g(y_{k'_{gd}k'}), \qquad (14)$$

where $s_{k'_{gd}}$ is the output of a sub-network with K hidden units, and $y_{k'_{gd}k'}$ is the k'th hidden output in the k'_{gd}th sub-network. We set the number of hidden units in a sub-network to K, which enabled learning of the sub-network. The learning equation for group dropout is

$$\boldsymbol{J}_{k'}^{(m+1)} = \boldsymbol{J}_{k'}^{(m)} + \frac{\eta}{N}\left(\sum_{l=1}^{K} g(d_l^{(m)}) - \sum_{l'=1}^{K} g(y_{l'}^{(m)})\right) \times g'(y_{k'}^{(m)})\boldsymbol{\xi}^{(m)}. \qquad (15)$$

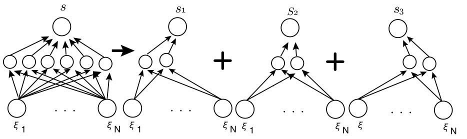

Fig. 10 Student divided into three sub-networks for ensemble learning. We assume teacher includes two hidden units.

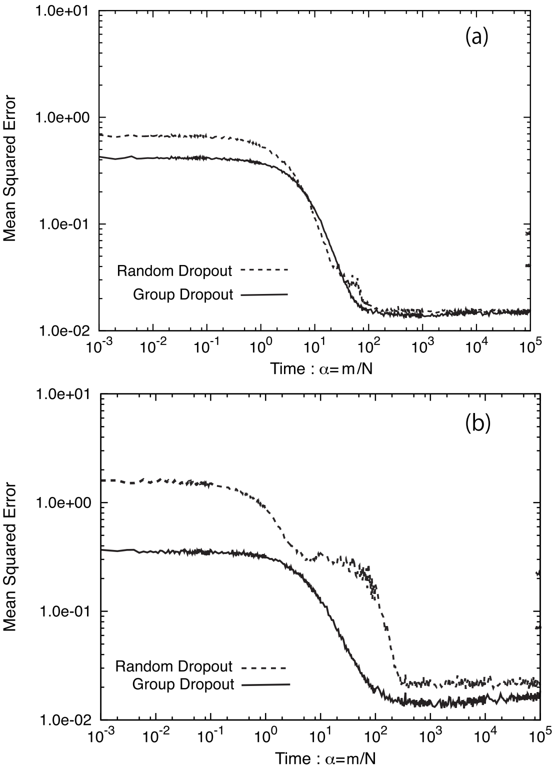

Fig. 11 Effects of proposed method (a) for $K' = 8$ and $K = 2$ and (b) for $K' = 30$ and $K = 2$ (average of ten trials).

Figure 11 shows the results obtained by taking the average of the results of ten trials. The training data and their targets were generated as described in Section 2. We generated $10 \times N$ training data instances and N test data instances in the same manner as described in Section 2. Each training data instance was frequently reused in the training phase. The number of input units was $N = 1{,}000$. Teacher included two hidden units. We used two students: (1) one included 8 hidden units, and (2) one included 30 hidden units. Case (1) will not lead to overfitting whereas case (2) will lead to overfitting. The effect of ensemble learning is small for case (1) and large for case (2).

As can be seen from Fig. 11, the residual errors for random dropout and group dropout converged to a low value, so there was no overfitting for the training data. Therefore, both random dropout and group dropout can work as a regularizer. When the number of hidden units was low (Fig. 11 (a)), the MSEs for group dropout and random dropout were identical. When the number was high (Fig. 11 (b)), the MSE for group dropout was smaller.

Next, we analyze the dynamic behavior of the proposed method. When the teacher has $\boldsymbol{B}_1$ and $\boldsymbol{B}_2$ and that the student

© 2017 Information Processing Society of Japan

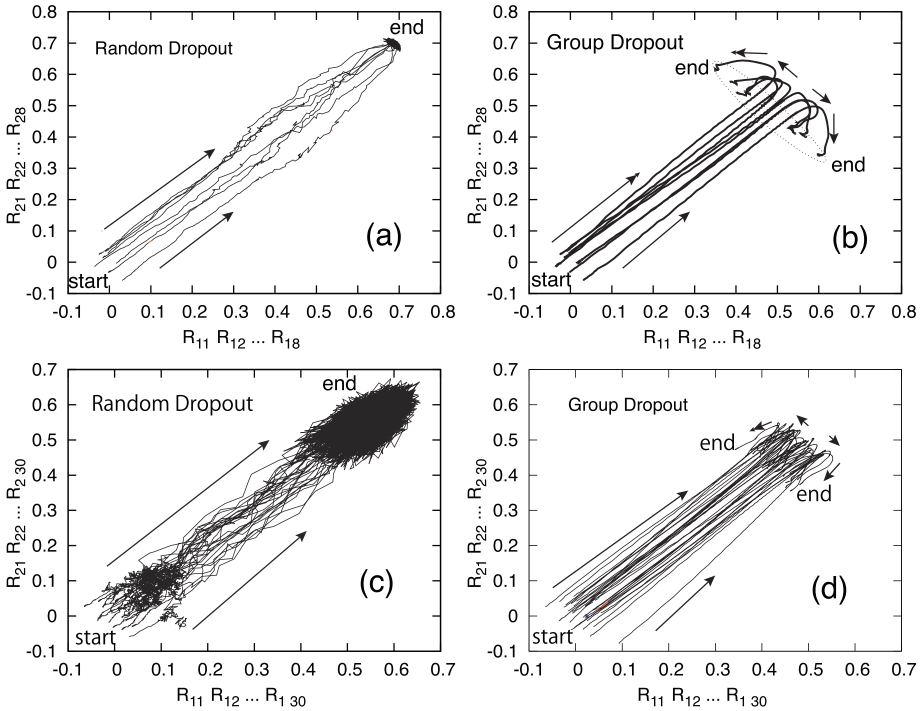

Fig. 12 Dynamic behaviors of overlap R during learning for random dropout and group dropout: (a) and (b) $K' = 8$; (c) and (d) $K' = 30$.

has $\boldsymbol{J}_1$ and $\boldsymbol{J}_2$, the same performance was achieved when $R_{11} = 1$, $R_{12} = 0$, $R_{21} = 0$ and $R_{22} = 1$ and when $R_{11} = 0$, $R_{12} = 1$, $R_{21} = 1$ and $R_{22} = 0$. This symmetry of the weight vectors occurred in each trail. Therefore, we cannot calculate the average dynamics of $R_{kk'}$. **Figure 12** shows typical behavior of overlap $R_{kk'}$ during learning for one of several trials using random dropout and group dropout. Figure 12 (a) and (b) show the results when $K' = 8$, and Fig. 12 (c) and (d) show the results for $K' = 30$. In Fig. 12 (a) and (b), the horizontal axis is R_{1x}, with x from 1 to 8. This axis shows the overlap of the teacher weight vector $\boldsymbol{B}_1$ and the student hidden weight vectors. The vertical axis is R_{2y}, with y from 1 to 8. This axis shows the overlap of teacher weight vector $\boldsymbol{B}_2$ and the student hidden weight vectors. In Fig. 12 (c) and (d), the x in R_{1x} on the horizontal axis runs from 1 to 30, and the y in R_{2y} on the vertical axis runs from 1 to 30. Learning started at label "start" and ended at label "end."

In Fig. 12 (a), thin lines start from "start" and converge near "end." During learning, the lines followed $R_{1x} = R_{2y}$. This means that the credits assigned to most of the weight vectors of the hidden units were similar. In Fig. 12 (b), the lines start from "start" and follow $R_{1x} = R_{2y}$ for a while but then turn in different directions near "end." This means that the credits assigned to the weight vectors differed. This broke the symmetry of the hidden unit weight vectors. Symmetry breaking of the hidden unit weight vectors occurs to escape from the singular point in the weight vectors space.

In Fig. 12 (c), the lines also started from "start" and converged into an area near "end." However, the lines for random dropout kept moving in a wider area than those in Fig. 12 (a). This mean that random dropout had more diversity when using many hidden units, thereby enhancing the ensemble effect. In Fig. 12 (d), the lines started from "start" and behaved similarly to those in Fig. 12 (b). Therefore, the residual error of group dropout tends to be smaller than that of random dropout when using many hidden units. Figure 12 (a) and (c) show that symmetry breaking did not occur. This suggests that random dropout does not fall into a singular point in the weight vector space.

Group dropout differs from random dropout in three ways. First, random dropout divides the student into two sub-networks whereas group dropout divides the student into more than two sub-networks. Second, random dropout randomly selects the hidden units to be neglected at each learning step whereas group dropout uses the same hidden units in each sub-network. Third, random dropout is the ensemble of learned and unlearned sub-networks whereas group dropout is the ensemble of only learned sub-netwroks.

Note that in group dropout, we assume that the number of hidden units in teacher is known. In general, the number of hidden units in teacher is not known. However, we can predict the number of hidden units in teacher by using model selection methods. By using the predicted number of hidden units in teacher, the proposed method may achieve performance similar to that described here.

5. Conclusion

We have presented our analysis of why random dropout can be regarded as ensemble learning. We first showed that the performance of dropout learning is similar to that of the SGD algorithm with L2 regularization despite their differing strategies. We then showed that random dropout can be regarded as ensemble learning except for when using a different set of hidden units in every learning iteration. This analysis clarified that using a different set of hidden units outperforms ensemble learning. We next presented our proposed method, group dropout, which divides the

© 2017 Information Processing Society of Japan

student into several sub-networks, each with the same number of hidden units included in the teacher and showed that its ensemble learning performance is better than that of random dropout when the number of hidden units in a sub-network is the same as that in the teacher. Future work includes clarifying the effect of averaging the outputs of the learned and unlearned hidden units and investigating group dropout when the number of hidden units in the sub-networks differs.

Acknowledgments The authors thank Professor Masato Okada and Assistant Professor Hideitsu Hino for their insights and advice. The authors also thank Professor Yutaka Sakai for his advice on comparing regularization performance between conventional dropout and the regularization method.

References

[1] LeCun, Y., Bengio, Y. and Hinton, G.: Deep learning, *Nature*, Vol.521, pp.436–444 (2015).

[2] Hinton, G.E., Osindero, S. and Teh, Y.W.: A fast learning algorithm for deep belief nets, *Neural Computation*, Vol.18, pp.1527–1554 (2006).

[3] Krizhevsky, A., Sutskever, I. and Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks, *Advances in Neural Information Processing Systems*, Vol.25, pp.1097–1105 (2012).

[4] Baldi, P. and Sadowski, P.: Understanding Dropout, *Advances in Neural Information Processing Systems*, Vol.26, pp.2814–2822 (2013).

[5] Freund, Y. and Schapire, R.E.: A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, *Journal of Computer and System Science*, Vol.55, pp.119–139 (1997).

[6] Hara, K. and Okada, M.: Ensemble Learning of Linear Perceptrons: On-Line Learning Theory, *Journal of the Physical Society of Japan*, Vol.74, pp.2966–2972 (2005).

[7] Miyoshi, S., Hara, K. and Okada, M.: Analysis of ensemble learning using simple perceptron based on online learning theory, *Physical Review E, American Physical Society*, Vol.71, 036116 (2005).

[8] Biehl, M. and Schwarze, H.: Learning by on-line gradient descent, *Journal of Physics A: Mathematical and General Physics*, Vol.28, pp.643–656 (1995).

[9] Saad, D. and Solla, S.A.: On-line learning in soft committee machines, *Physical Review E*, Vol.52, pp.4225–4243 (1995).

[10] Hara, K., Saitoh, D. and Shouno, H.: Analysis of Dropout Regarded as Ensemble Learning, *ICANN 2016, Part II, LNCS*, Vol.9887, pp.72–79 (2016).

[11] Saitoh, D., Kondou, T. and Hara, K.: Proposal of novel dropout method and its analysis of dynamic property, *Technical Report of IEICE*, NC2015-67, pp.55–60 (2016-1). (in Japanese)

[12] Hara, K., Saitoh, D., Kondou, T., Suzuki, S. and Shouno, H.: Group Dropout Inspired by Ensemble Learning, *ICONIP 2016, Part II, LNCS*, Vol.9948, pp.66–73 (2016).

[13] Bishop, C.M.: *Pattern Recognition and Machine Learning*, Springer (2006).

[14] Wager, S., Wang, S. and Liang, P.: Dropout Training as Adaptive Regularization, *Advances in Neural Information Processing Systems*, Vol.26, pp.351–359 (2013).

Kazuyuki Hara received a B.Eng. and an M.Eng. degrees from Nihon University in 1979 and 1981 respectively and a Ph.D. degree from Kanazawa University in 1997. He was involved in NEC Home Electronics Corporation from 1981 until 1987. He joined to Toyama Polytechnic College in 1987 where he was a lecturer. He joined Tokyo Metropolitan College of Technology in 1998 where he was an associate professor and became a professor in 2005. He became a professor at Nihon University in 2010. His current research interests include statistical mechanics of on-line learning.

Daisuke Saitoh received a B.Eng. and an M.Eng. degrees from Nihon University in 2014 and 2016 respectively. He is involved in TSP Co., Ltd. from 2016 where he is an Technical engineer. His interest in research includes the online learning.

Satoshi Suzuki was born in 1993. He received a B.E. degree from University of Electro-Communications in 2015. He is studying Computer Vision and artificial neural network. And he is a member of IEEE, IPSJ, and JNNS.

Takumi Kondou received a B.Eng. and an M.Eng. degrees from Nihon University in 2015 and 2017 respectively. His interest in research includes the online learning.

Hayaru Shouno was born in 1968. He received a M.E. and a Ph.D. degrees from Osaka University in 1994 and 1999. His current research interest is in image processing and artificial neural network. He is a member of IEEE, IEICE, and IPSJ.

© 2017 Information Processing Society of Japan