International Journal of Computer Information Systems and Industrial Management Applications.
ISSN 2150-7988 Volume 8 (2016) pp. 067–076

A Study on Visualizing Feature Extracted from Deep Restricted Boltzmann Machine using PCA

Koki Kawasaki[1], **Tomohiro Yoshikawa**[1] **and Takeshi Furuhashi**[1]

[1]Nagoya University, Graduate School of Engineering,
Furocho, Chikusa ward, Nagoya city, Aichi prefecture, Japan
kawasaki@cmplx.cse.nagoya-u.ac.jp, yoshikawa@cse.nagoya-u.ac.jp, furuhashi@cse.nagoya-u.ac.jp

Abstract: **P300 speller is a system that allows users to input letters using only electroencephalogram (EEG). A component called P300 is used to interpret the EEG in P300 speller. In order to achieve high performance in P300 speller, achieving high performance of P300 detection is essential. However, EEG waveforms are strongly dependent on the conditions of subject and/or environment, so it is not easy to detect P300 precisely. In this study, deep neural network using restricted boltzmann machine, which became famous by its high performance, is used to detect P300. It is expected that it also shows high performance for complex EEG waveforms. The experimental result shows that deep neural network was able to detect P300 better than the existing method (stepwise linear discriminant analysis). Furthermore, this study refers to the learned feature by deep restricted boltzmann machine. We can see that deep restricted boltzmann machine learns the feature extracted from the EEG waveforms correctly to detect P300 which led to the high performance.**

Keywords: Restricted boltzmann machine, Deep neural network, P300 detection, Feature extraction, Visualizing feature, Principal component analysis

I. Introduction

Brain-computer interfaces are the systems that allow users to control devices just by their own thought [1]. P300 speller [2] is one example for brain-computer interfaces, which allows users to input letters using only electroencephalogram (EEG). Fig. 1 shows an example of the interface of P300 speller. Each row and column flashes each by each. By analyzing the EEG for each flash, the letter to be input is discriminated. Brain-computer interface like P300 speller is very helpful for people who cannot move their muscles by theirselves, *e.g.* Amyotrophic Lateral Sclerosis (ALS) patients [3]. P300 speller can be an important device to make it possible to communicate with other people.

In P300 speller, a component called P300 is used to interpret the EEG. P300 is one of the Event Related Potentials (ERPs), which can be confirmed by the enhanced component appearing in the EEG. It typically appears about 300ms after the infrequent stimuli which is presented to the subject within sequences of frequent stimuli. In order to achieve high performance in P300 speller, achieving high performance of P300 detection is essential. What is inconvenient in EEG waveforms is that they are strongly dependent on the conditions. For example, they show different feature when a user feels sleepiness [4]. Thus, learning P300's feature correctly is important. In P300 speller, StepWise Linear Discriminant Analysis (SWLDA) is one famous method to detect P300. SWLDA is a linear discriminant which selects variables to be inputted to make suitable model. SWLDA is thought to perform better than Support Vector Machine (SVM); a famous discrimination method [5].

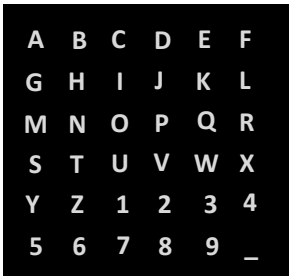


Figure. 1: Example of P300 speller interface

In this study, Deep Neural Network (DNN) is used to detect P300. DNN is a neural network with multi-layer. The word "Deep Learning" is used to represent the learning methods for DNN. DNN is gaining more and more attention from researchers recently [6]. In the research fields of image processing, DNN is well known by having high discrimination performance [7]. It was a big impact to significantly improve the discrimination result (more than 10% in error rate) for the image used in the ImageNet LSVRC-2010 contest. Le *et al.* made a grandmother cell of human face and cat from huge numbers of data [8], which was also a big impact in the region of machine learning.

Nowadays, many researchers are studying DNN in various research fields expecting for its high performance as in the image processing. Noda *et al.* used DNN in audio-visual speech recognition field [9]. DNN filtered out the effect of noise and learned the feature of image from the mouse area well to predict the words. Anderson *et al.* were trying to discover interesting patterns in climate data by using DNN [10]. Lenz *et al.* used DNN for robots to determine where to grasp an object from images [11]. In most researches, using DNN made it possible to achieve higher performance than the state-of-the-art methods.

There are many existing methods for DNN, *e.g.* restricted boltzmann machine [12, 13], auto-encoder [14], convolutional neural network, recurrent neural network, and so on. The most remarkable characteristic of DNN is that it can learn the representation of input data automatically, which was not easy before the appearance of DNN. This means the feature of P300 can be expected to be learned automatically, which can reduce the need of specialist of EEG data in P300 detection task, considering that it is hard to choose features from the data by human hands. There is not many reports using DNN for P300 detection. Cecotti and Gräser used convolutional neural network for P300 detection and achieved high performance [15].

However, there are demerits in the exchanges of the merits in using DNN. One of them is that the training time is much longer and the calculation cost is much higher than the existing methods due to the deepness of networks, which also makes it difficult to tune and optimize the parameters of DNN. Another demerit is that it is not clear how DNN learns the feature from data. This means that the procedure of discrimination is in a black box.

To understand the procedure of discrimination of DNN, many researches are done. Le made a grandmother cell to understand the feature extracted from the DNN [8]. This uses an optimization method to find the grandmother cell. The input which minimizes the sum of the error of reconstruction and the spasity value is called as the grandmother cell in that paper. This method reveals what the network learned most precisely and cannot reveal what each node learned.

Erhan *et al.* introduced a mothod called "maximizing the activation" and "sampling from a unit" [16]. The former method is another optimization method. It find the input to maximize a paticular node's value, and the obtained input is thought to be the input to activate the node the most. In this method, we can know what each node in particular layer learned, but some time it will give us only few information. The latter method will be explained in the next section.

Semak *et al.* used a heatmap to evaluate what DNN learned [17]. Three methods of calculation for obtaining the heatmap were introduced. In the paper, the heatmap of "layerwise relevance propagation method" seems to be much noiseless heatmap. The paper insisted that "heatmaps may be useful for assessment of neural network."

Yosinski *et al.* made two tools for visualization [18]. One tool was to see how the activation changes among different input data. This can lead to make worthy intuitions for how neural network works. However, seeing individual data will take time, especially when the network learned feature from large number of data. The other tool was to visualize through regularized optimization in the input space as similar to [16]. Four regularized functions were used and enabled more interpretable visualizations. The paper also pointed out that the feature visualization became more complex in the higher layers.

In this study, we employ a visualization approach using Principal Component Analysis (PCA) and extending "sampling from a unit" method [16] to understand what kind of feature was learned by DNN. This paper will concentrate on the use of deep Restricted Boltzmann Machine (RBM) for P300 detection. We will observe the performance of deep RBM comparing with SWLDA. Then the learned feature by deep RBM will be observed. We will also use PCA to study the relationship between the deepness of the network and the performance of the detection.

II. Methods

A. RBM

RBM is a type of learning methods for neural network. It can be said as one of the deep learning methods. RBM can learn the representation from data[19]. RBM is a generative model which generates the probability of data, which gives the difference from auto-encoder; a similar algorithm with RBM. RBM has three layers. The first layer is the input layer, also called as the visible layer, which the data is inputted to. The second layer is the hidden layer, which can be said as the representation of the previous layer. The third layer is the output layer, which has the same number of nodes with the input layer. As the learning proceeds, the output layer is more likely to have close value with the input layer. An example of RBM is shown in Fig. 2.

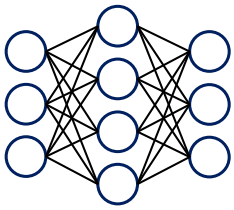


Figure. 2: Structure of three layers RBM

The weight matrix between the input layer and the hidden layer is set as $\boldsymbol{W}$, and that between the hidden layer and the output layer is set as $\boldsymbol{V}$. It is usually restricted as $\boldsymbol{V} = \boldsymbol{W}^T$ for calculation easiness. The bias vectors of the hidden layer and the output layer are set as $\boldsymbol{b}$ and $\boldsymbol{c}$, respectively. Using these parameters; $\boldsymbol{W}$, $\boldsymbol{b}$ and $\boldsymbol{c}$; the output $\boldsymbol{z}$ can be calculated from the input $\boldsymbol{x}$ as eq.(1), where $f(x)$ is the activation function (sigmoid function is applied in this paper).

$$\boldsymbol{z} = f(\boldsymbol{W}^T \cdot f(\boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}) + \boldsymbol{c}) \tag{1}$$

In RBM, the value of each node in all layer is chosen to be 0 or 1. They take continuous value between 0 and 1 in the calculation, and each value can be thought to be the probability if the node's output becomes 1. To calcuate the probability appropriately, parameters; $\boldsymbol{W}$, $\boldsymbol{b}$ and $\boldsymbol{c}$ must be determined. Parameters can be learned by maximizing log-likelihood [12]. This maximization of log-likelihood has abundant calculation, so approximation is used to calculate. The approximated algorithm is well known as contrastive divergence method [20].

To make it able to input continuous value to RBM, modifying the model must be done. Though it is said to be hard to modify the model keeping the learning performance, studies are done to input continuous value [21, 22]. In this paper, we use a very simple modification; to change the activation function to identity function, as introduced in [23].

The learning of the parameters in RBM is done by unsupervised learning. This means that RBM can extract the feature of data without any supervision.

B. Deep RBM

Deep RBM can be made by just adding new RBM on the top of the previous RBM's hidden layer. For example, when the structure in Fig. 3 is the network to be learned, the learning will be done by doing three steps of learning as shown in Fig. 4. The first step learns the weight between the first two layers and biases of the first two layers. The learning of these parameters are done by RBM introduced in the previous section. The first RBM with 3 nodes as the input layer, 4 nodes as the hidden layer, 3nodes as the output layer is made. After the parameters are learned, $\boldsymbol{W}$ will be the weight between the first two layers in Fig. 3, $\boldsymbol{b}$ and $\boldsymbol{c}$ will be the bias for the second and first layer, respectively. Then the next step is to learn the parameters of the next two layers in the same way. This way of learning (learning two layers at a time) is called as greedy layer-wise learning [12]. In RBM, layers extract the representation from the data of previous layer. Thus in deep RBM, the highest layer can be interpreted as the "ultimate representation" which is extracted from input data.

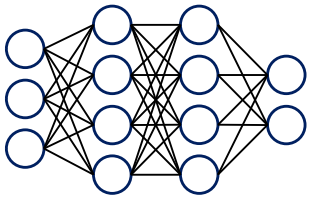


Figure. 3: Deep RBM structure

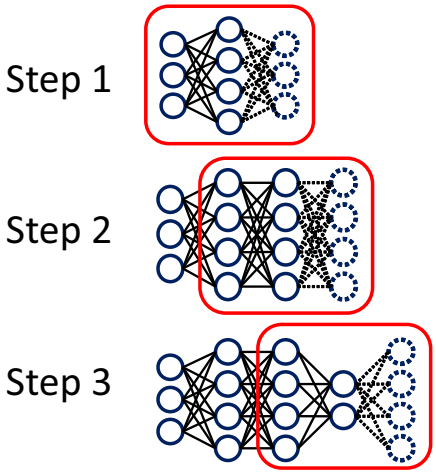


Figure. 4: Three steps of learning

C. P300 detection

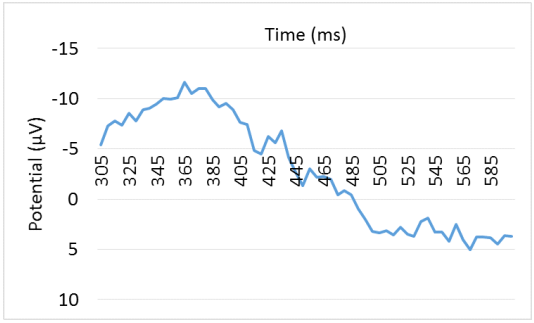
Deep RBM can extract the feature from the data, but cannot discriminate data. Thus, by using the representation extracted from RBM as the input, supervised learning can be done to discriminate the data. This supervised learning is usually conducted by the neural network method, but it can be also conducted by other discrimination method such as SVM [24]. In this paper, back-propagation; a famous method to learn neural network, is used for the supervised learning.

D. Visualization of Extracted Feature

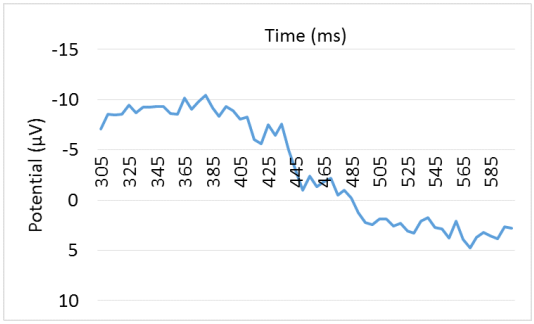
It is not clear how DNN extracts the feature from input data. However, we can visualize what DNN has extracted, which will help us to understand the DNN's behavior. In this paper, we use a similar way to "sampling from a unit" [16] to visualize the extracted feature. "Sampling from a unit" is a way to understand the extracted feature by seeing it in the input space. For example, seeing the relationship between the input space and the weights, it makes it easier to understand the extracted feature. The procedure for "sampling from a unit" is described below. Though the word "unit" is used in [16], "node" is used instead in this paper.

1. Choose a layer to be visualized.
2. Choose a single node from the chosen layer.
3. Make a vector constraining the element corresponding to the chosen node to 1, and otherwise 0.
4. Back-propagate the constrained vector to the input layer. The acquired vector by this back-propagation is called the result vector.

The result vector of "sampling from a unit" can be considered as the typical input to activate the chosen node, which means that the chosen node will take high value when a data like the result vector is inputted. However, extracted feature should be mostly represented with plural nodes. Thus, just sampling "one" node may give us little information, or the result vector may be too abstract to understand. As an example, Fig. 5 shows the result for different two selected nodes. The result looks quite alike. If all nodes in the same layer shows similar result, it would be hard to tell what kind of feature the network learned.



(a) Example 1



(b) Example 2

Figure. 5: Results of "sampling from a unit" in layer 5

Thus we extend "sampling from a unit" method to sample "plural" nodes. In choosing plural nodes, the problem is how to choose them. In this study, we use Principal Component Analysis (PCA) as the criterion of choice. The procedure for visualizing the extracted features with plural nodes is described below.

1. Choose a layer to be visualized.
2. Conduct PCA to the matrix of output values of the chosen layer by the input data, *i.e.* the matrix size is (input data X nodes in the chosen layer).
3. Plot all nodes of the chosen layer based on the PC loadings. Plotting are done with x-axis (PC1) and y-axis (PC2) as the two PCs chosen in step 3.
4. Choose the characteristic nodes which are activated when the average P300 waveform of raw data is inputted.
5. Make a vector constraining the element corresponding to the chosen nodes to 1, and otherwise 0.
6. Back-propagate the constrained vector to the input layer and acquire the result vector.

III. Experiments

A. EEG Dataset

The EEG dataset used in this experiment was collected from 10 subjects. The subjects were in thier 20's and healthy persons. The EEG was collected as follows.

> A display was set 1m far from the subject. The subject sat on a chair and was told to stare at the display. In the display, figures were presented at the center in a fixed frequency; figures appeared for 400ms after 400ms of black out. Two types of figures were presented. One was the target figure, which subject was told to click the mouse when it was presented. The other was the nontarget figure, which subject was told to do nothing.

When the target figure was presented, P300 was expected to appear, and when nontarget, P300 was not expected to appear.
9 electrodes were used to collect the EEG data. 400 times of figure presentations were done to make the dataset. The ratio of target to nontarget was set to 1:9. As for the training dataset, EEG data for 300 times of figure presentations were used, fixing the frequency of the target data to 10%. 9 electrodes were regarded as different data on each figure presentation, which expands the training dataset to 2,700 data. As for the test dataset, EEG for 100 times figure presentations were used, fixing the frequency of the target data to 10%. Same with the training data, 9 electrodes were used as different data, and the testing dataset was 900 data.

B. Settings for DNN

In the experiment, DNN with 6 layers were employed. The number of nodes for each layer were set as 60, 100, 60, 60, 100, and 1, respectively. These were set arbitrarily, *i.e.* there were not much meaning. The learning rates for the feature extracting step and P300 detection step were both set to 0.10. The number of iterations for feature extracting step and P300 detection step were set to 2,000 and 10,000, respectively, again with not much meaning. The output layer's node was set to be 1 for P300 data. This means the output layer outputs the label as a real number for the input data. In other words, it is learned to output 1 when a target data (P300 data) is inputted, and output 0 when a nontarget data (nonP300 data) is inputted. In the following of this paper, the output value of the network will be called as "score" for the input data.

C. Accuracy Evaluation for Each Layer

In DNN, each layer's outputs, except for the input layer and output layer, are assumed as the extracted representation from the previous layer. The higher layer it goes, the more abstract representation, which can be regarded as the feature, are thought to be extracted by DNN [8]. So we studied the performance of the extracted representation for each layer. By making an output layer on the top of the chosen layer and learning the added parameters, *i.e.* conducting the supervised learning step using the representation extracted from the chosen layer, we can calculate the score for each input data for each layer. By conducting this experiment, we can find out how the performance changes as the extracted representation changes. Precision, recall, accuracy and F-measure were employed as the evaluation indexes. All four indexes take value between 0 and 1. Higher value of each index means higher performance. The indexes were calculated using formulas from eq.(2) to eq.(5) based on the values shown in Table 1.

$$precision = \frac{TP}{TP+FP} \tag{2}$$

$$recall = \frac{TP}{TP+FN} \tag{3}$$

$$accuracy = \frac{TP+TN}{TP+FP+TN+FN} \tag{4}$$

$$F-measure = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{5}$$

Table 1: Explanation for evaluation indexes

		True state	
		Target (P300)	Nontarget (nonP300)
Predicted State	P300	TP	FP
	NonP300	FN	TN

To compare the result of DNN with other methods, we conducted an experiment using SWLDA as the P300 detection method. SWLDA used raw EEG data to discriminate. Thus the dataset for SWLDA was the same with that for DNN. The parameters for SWLDA were set as the default of "MATLAB," a software for numerical analysis.
We also compared the performance of DNN using RBM with DNN using only back-propagation for the learning method. The setting for this network was set as the same of DNN using RBM explained in III-B. This comparision will examine the effect of using RBM as the learning algorithm, not deep back-propagation.

IV. Results and discussions of performance

First, we will see the result of the performance for one subject. Then we will see the performance of all 10 subjects in the last subsection.

A. DNN using RBM

The average and the standard deviation of the score for target test data and nontarget test data are shown in Table 2. The target data was learned to output 1, so the average of the score for target data is expected to be close to 1. However, the result shows that the average of target data is not close to 1. This is thought to be because of the inconsistency of the label. The label was conferred based on the presented figure. So when the target figure was presented, the label for that data was set as "target," which means P300 is expected to appear. If the subject happened to miss the appearance of target figure, P300 would not appear in the EEG waveform, which will cause the inconsistency of the label; the data is labeled as P300 but it is actually nonP300 data. This will make it hard to achieve high average score in P300 data. However, the difference of the score was large enough to detect P300 in many data.

Table 2: Average and standard deviation of score using RBM

	Target	Nontarget
Average	0.378	0.060
Standard deviation	0.242	0.128

A threshold has to be set in order to detect P300 with the score calculated in DNN. If the score is higher than the threshold, the data will be regarded as P300. The performance will change depending on the threshold, so the threshold which achieves the best performance in the training data was searched based on the F-measure. The best F-measure for each layer and its threshold are shown in Table 3. In this table, the best F-measure is calculated using the training data. The layer number means which layer's representation was used to detect P300. In Table 3, the best F-measure is increasing as the layer gets higher, which means the representation extracted by DNN is better in the higher layer. However, the F-measure is relatively low even by the best one. This is also thought to be because of the inconsistency of the label as explained above.

Table 3: Best F-measure for each layer

Layer number	Layer 2	Layer 3	Layer 4	Layer 5
Best F-measure	0.486	0.529	0.578	0.591
Corresponding threshold	0.230	0.260	0.260	0.300
Corresponding precision	0.409	0.474	0.504	0.531
Corresponding recall	0.600	0.600	0.678	0.667
Corresponding accuracy	0.873	0.893	0.901	0.908

B. SWLDA

The average and the standard deviation of the score for target test data and nontarget test data by SWLDA are shown in Table 4. The same word "score" is used in this subsection, but we have to be sure that the score of DNN and that of SWLDA have different range; DNN's score is a numeric value between 0 and 1, while SWLDA's score can take any real number. The average target score differs from the average nontarget score, but the standard deviation was large in both two data as shown in Table 4.

Table 4: Average and standard deviation of score using SWLDA

	Target	Nontarget
Average	213.4	-14.2
Standard deviation	366.8	229.1

The best F-measure result of P300 detection using SWLDA is shown in Table 5. The best F-measure using SWLDA was lower than that by DNN using RBM. One reason for this result is because of the largeness of the standard deviation of the score, which made it hard to discriminate the data. As the result shows, we can say that DNN using RBM learned the feature of P300 better than SWLDA. SWLDA cannot make complex boundary line because it is a linear discrimination method. However, EEG data are very complex waveforms. The simpleness of SWLDA may have made it hard to detect P300 in the same condition as DNN. Furthermore, considering SWLDA discriminated with raw data as input, the extracted representation in DNN seems to make it easier to discriminate, compared with raw EEG data.

Table 5: Best F-measure using SWLDA

	SWLDA
Best F-measure	0.392
Corresponding threshold	322(μV)
Corresponding precision	0.398
Corresponding recall	0.389
Corresponding accuracy	0.880

C. DNN using back-propagation

The average and the standard deviation of the score for target test data and nontarget test data by DNN using back-propagation are shown in Table 6. The result shows that the target score and nontarget score were very close.

Table 6: Average and standard deviation of score using back-propagation

	Target	Nontarget
Average	0.09999	0.09998
Standard deviation	5.9×10^{-6}	6.2×10^{-6}

The best F-measure of DNN using back-propagation is shown in Table 7. As the result shows, though there was no significant difference in target score and nontarget score, the P300 detection result was not as poor as SWLDA. This can be explained by its smallness of the standard deviation.

The standard deviation for both target data and nontarget data were smaller than the difference of the average score for the two, which were enough to discriminate them. However, DNN using RBM in higher layers detected P300 better. This would be the effect of using RBM as representation learning. DNN using back-propagation did not extract feature from the data but learned the weights and biases by the supervised learning. However, as it is said widely, deep back-propagation might end up into localized solution. By learning representation of the data beforehand in DNN using RBM, that problem could be solved. Thus DNN using RBM outperformed DNN using back-propagation.

Table 7: Best F-measure using back-propagation

	Back-propagation
Best F-measure	0.570
Corresponding threshold	0.100
Corresponding precision	0.573
Corresponding recall	0.567
Corresponding accuracy	0.914

D. Results of all 10 subjects

The result of performance for all 10 subjects is shown in Table 8. We compared the result of DNN using RBM with SWLDA. As Table 8 shows, the performance of DNN is mostly better than that of SWLDA; in 8 subjects out of 10, DNN using RBM showed higher F-measure. The last row of Table 8 shows the average F-measure for DNN and SWLDA. The average of DNN is higher, thus it can be said that DNN tends to achieve better performance. DNN first extracts representation from the data, but SWLDA does not. Furthermore, the deepness of DNN helped learn efficient representation from the data *i.e.* the feature of the data. These advantages of DNN using RBM helped achieve better performance.

Table 8: Best F-measure for all 10 subjects

	SWLDA	DNN using RBM
sub1	0.354	0.523
sub2	0.513	0.436
sub3	0.192	0.282
sub4	0.192	0.336
sub5	0.425	0.394
sub6	0.349	0.495
sub7	0.283	0.307
sub8	0.080	0.130
sub9	0.145	0.199
sub10	0.151	0.156
average	0.268	0.326

V. Results and discussions of visualizing feature learning

A. Results of visualization for one subject

The visualization was conducted as it is introduced in II-D. Table 9 shows the PC score for P300 data and nonP300 data by conducting PCA to the matrix of the output values by the input data for layer 2 to layer 5, *i.e.* step 2 in the II-D "sampling from plural nodes." In Table 9, PC is put in the order of descending contribution ratio. The range of the shown PC is determined by the range between PC1 and the first PC to exceed 80% in cumulative contribution ratio (CCR), except for layer 2, where the CCR first exceeded 80% in PC38. In Table 9, the PC score of P300 and nonP300 data are different in some PCs. The difference seems to be relatively larger compared with other PCs especially in PC1 and PC2.

The result for plotting all nodes in layer 5 based on the PC loadings, *i.e.* step 4 in II-D is shown in Fig. 6. In Fig. 6(a), the probability of activation for the average of P300 raw data is shown with the gradation of color. When the color of each node is darker, it means the node is more likely to be fired when the average of P300 raw data is inputted. We can see that the average of P300 raw data fires a particular node region. When we compare the result with the output of average of nonP300 raw data shown in Fig. 6(b), we can also see the difference of fired nodes between the average of P300 and nonP300 raw data. There is a region which is fired when both P300 and nonP300 data is inputed. There is also a region which is not fired when either P300 or nonP300 data is inputed. These nodes may be a meaningless nodes, which holds no information in terms of the difference between the P300 and nonP300 data. Omitting these nodes and building a new structure of neural network may lead to a better model, but this is not studied in this paper, and will be the future work.

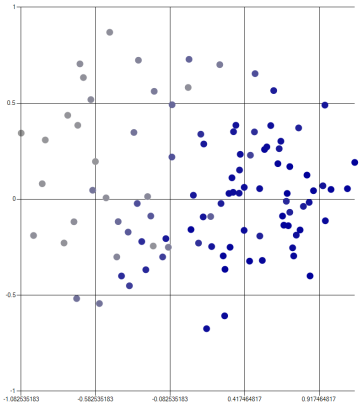
The characteristic nodes to back-propagate were subjectively chosen from the result shown in Fig. 6(a). For example, for P300 data in Fig. 6(a), the region of rightbottom was selected, and for nonP300 data in Fig. 6(b), the region of left was selected. From the map of PCA, we can say that the closer the nodes are, the more related they are. This means if two nodes are mapped close to each other in the map of PCA, the two nodes will show similar charactaristics. Thus even some nodes in the selected region is not fired, it should be interpreted as miss-unfire; it should have been fired. To correct the miss-unfire, we chose the region of nodes instead of individual nodes.

The eventual results of conducting the procedure explained in section II-D is shown in Fig. 7. In the layers from layer 2 to layer 5, the enhanced component of P300 can be confirmed from the extracted P300 feature. It may not be a significant difference, but the enhanced component is emphasized as the layer gets higher. The peak latency of P300 data seems not to change even when the layer goes higher; around 425ms. From these results, even though the F-measure was not indeed high, we can verify that DNN using RBM extracted the feature of P300 as expected.

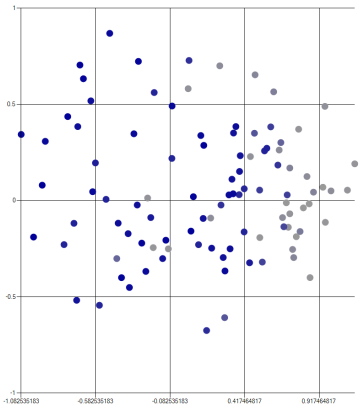
To compare the result with "sampling from a unit," we did another experiment. We chose layer 5 and "sampled from a unit" for all nodes in layer 5. Two of the results, in which the node was selected randomly, are shown in Fig. 5. Two waveforms in Fig. 5 look quite similar. Same kinds of results were acquired in all nodes in layer 5, *i.e.* all results in layer 5 looked alike. This is thought to be because the effect of a single node was too small to explain the feature extracted from the input data. Thus it is shown that "sampling from

Table 9: PC score for P300 data and nonP300 data in each layer

	Layer 2		Layer 3		Layer 4		Layer 5	
	P300	NonP300	P300	NonP300	P300	NonP300	P300	NonP300
PC1	-0.517	1.685	3.233	2.725	-1.867	-1.493	2.598	2.566
PC2	2.949	0.714	2.369	0.026	-1.750	0.463	3.353	0.682
PC3	0.152	0.151	0.705	0.808	0.166	0.285	0.695	0.893
PC4	0.325	0.388	-0.025	0.090	0.004	-0.134	-0.211	-0.071
PC5	0.291	0.075	-0.254	-0.152	0.446	0.368	0.428	0.387
PC6	0.071	0.019	0.155	0.087	-0.269	-0.265		
PC7	-0.239	-0.182	0.178	0.112				
PC8	-0.070	-0.009	0.178	0.115				
PC9	0.238	0.096						
PC10	0.207	-0.048						



(a) Gradation by average P300



(b) Gradation by average nonP300

Figure. 6: Plotting nodes in layer 5 based on PC loading

plural units" can give more information than "sampling from a unit" especially in higher layers.

B. Results of visualization for all 10 subjects

The visualization results of other subjects will be presented in this subsection. First we will show the result for sub8. This subject had the lowest performance as Table 8 shows. The results of visualization is shown in Fig. 8. From this figure, we can see that the P300 representation has an enhanced component in layer 2 to layer 4. The peak latency seems to be arround 420ms. However, in layer 5, the representation of P300 is different from others. The peak latency seems to be arround 500ms, and the potential of 550 - 600ms is higher than the representation extracted from other layers. These characteristics of P300 are not as expected. Thus this layer 5 caused the low F-measure in this subject; the representation of P300 was not learned properly.

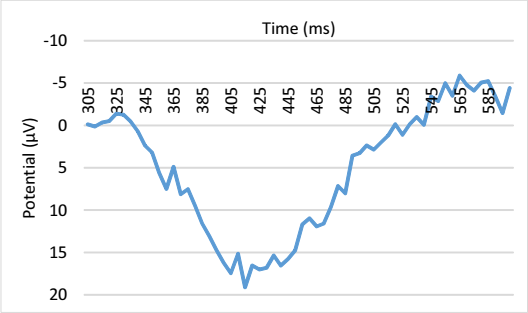
Next we will show the result of sub3 in Fig. 9. We can see that it is hard to detect P300 in this subject. Fig. 10 shows the mean waveform of target/nontarget data for sub3. It can be said that P300 seems not to appear in all target data, or the P300 is weak for this subject. From the visualization results, although the peak potential is relatively low, an enhanced component similar to P300 can be confirmed in all layer's representation. This means that DNN using RBM learned feature of P300 from little data. This made it possible to detect P300 better than SWLDA.

VI. Conclusion

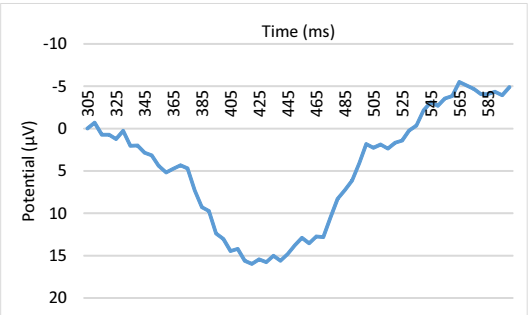
In this paper, the effects of using DNN for P300 detection was studied. The experiment comparing the best F-measure showed that DNN using RBM could detect P300 better than SWLDA and DNN using back-propagation. This result is thought to be because RBM extracted feature from the data properly, which made the detection easier. The problem of data used in the experiment was that the label for the prepared data was not always correct, which made it hard to achieve high performance results in the experiment. The visualization of extracted representation showed that DNN extracted the feature of P300 as expected in many subjects. However, in some subjects, the feature was not extracted as good as other subjects. In those subjects, the performance seemed to be relatively lower.

One future work is to use larger dataset for the investigation, because DNN is thought to have higher performance when the training dataset is large enough. Making dataset larger also means to make training dataset with plural subjects. If this can be done, global P300 feature for many people *i.e.* grandmother cell for P300, may be able to be extracted.

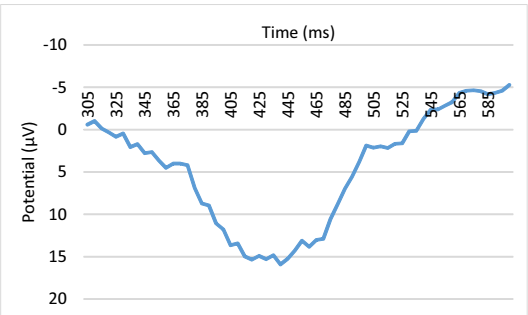
Another future work is to optimize the parameters used in DNN, because the parameters in this paper, such as the number of nodes in all layers, were set arbitrarily. An approach of the optimization for parameters may be figured by using the result of visualization, because the visualization is one



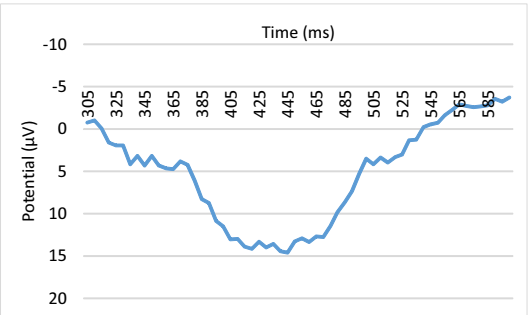
(a) Layer 2



(b) Layer 3

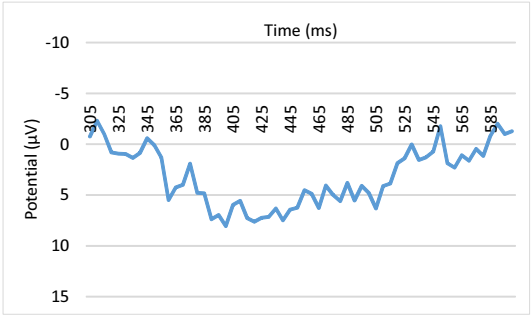


(c) Layer 4

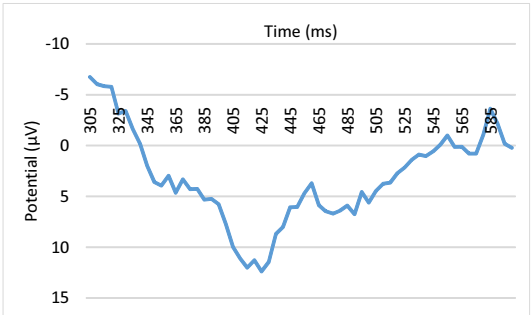


(d) Layer 5

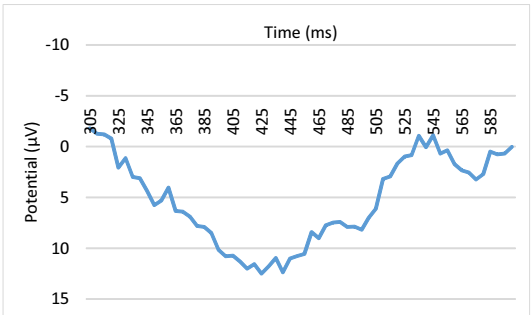
Figure. 7: Extracted P300 features from each layer for sub1



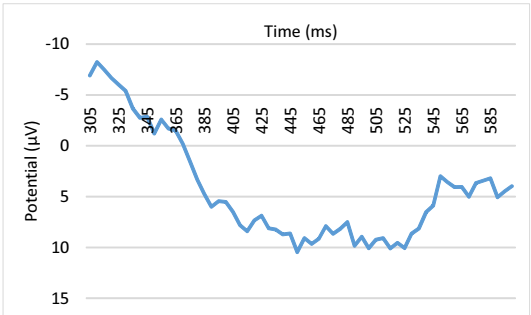
(a) Layer 2



(b) Layer 3

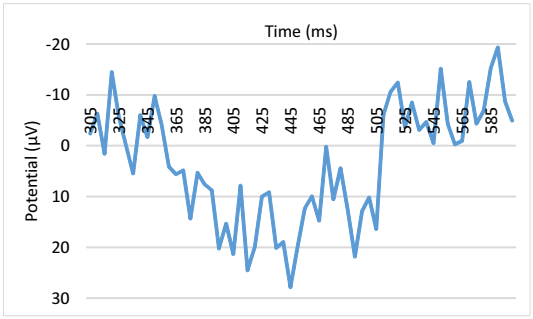


(c) Layer 4

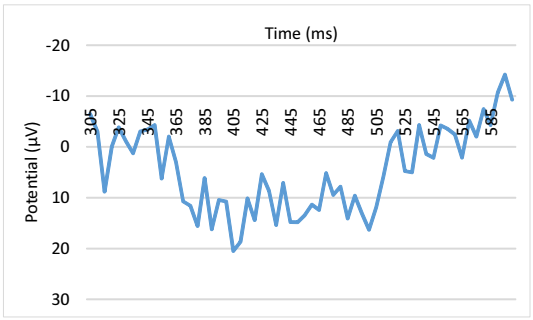


(d) Layer 5

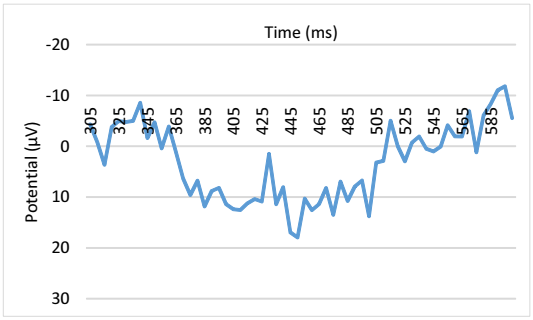
Figure. 8: Extracted P300 features from each layer for sub8



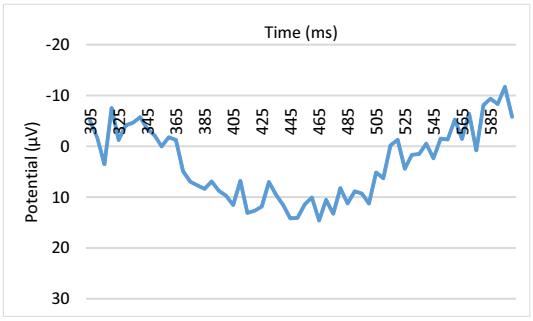
(a) Layer 2



(b) Layer 3



(c) Layer 4



(d) Layer 5

Figure. 9: Extracted P300 features from each layer for sub3

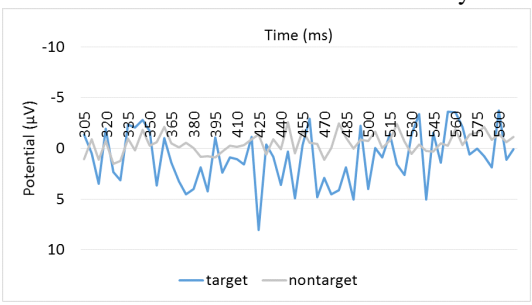


Figure. 10: Mean waveform of sub3

of the ways to interpret the process of learning in DNN. In proper learning, the region of target data and the region of nodes which is likely to be fired when P300 data is inputted are thought to be close to each other, as in Fig. 11. Thus algorithms to take these visualization results into account for parameter optimization can be the next step of this study.

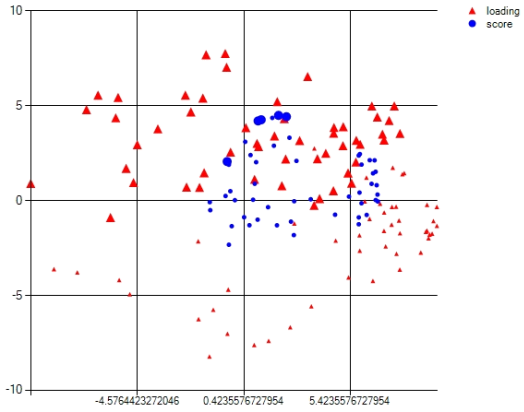


Figure. 11: Map of PC scores and PC loadings

Another future work is to use the information of electrodes to detect P300. In this paper, the 9 electrodes used in the collected data was regarded as different and independent data. However, there should be some kinds of interaction between electrodes, so taking that interaction into account will give better performance.

Acknowledgments

This work was supported by “Center of Inovation (COI) program,” Japan Science and Technology Agency (JST).

References

[1] J. R. Wolpaw, N. Birbaumer, D. J. McFarland, G. Pfurtscheller, and T. M. Vaughan. Brain–Computer Interfaces for Communication and Control. *Clinical Neurophysiology*, 113(6):767–791, 2002.

[2] L. A. Farwell and E. Donchin. Talking off the Top of Your Head: Toward a Mental Prosthesis Utilizing Event-Related Brain Potentials. *Electroencephalography and Clinical Neurophysiology*, 70(6):510–523, 1988.

[3] D. Kaub-Wittemer, N. von Steinbüchel, M. Wasner, G. Laier-Groeneveld, and G. D. Borasio. Quality of Life and Psychosocial Issues in Ventilated Patients with Amyotrophic Lateral Sclerosis and Their Caregivers. *Journal of Pain and Symptom Management*, 26(4):890–896, 2003.

[4] J. Harsh, U. Voss, J. Hull, S. Schrepfer, and P. Badia. ERP and Behavioral Changes During the Wake/Sleep Transition. *Psychophysiology*, 31(3):244–252, 1994.

[5] D. J. Krusienski, E. W. Sellers, F. Cabestaing, S. Bayoudh, D. J. McFarland, T. M. Vaughan, and J. R. Wolpaw. A Comparison of Classification Techniques for the P300 Speller. *Journal of Neural Engineering*, 3(4):299–305, 2006.

[6] M. Längkvist, L. Karlsson, and A. Loutfi. A Review of Unsupervised Feature Learning and Deep Learning for Time-Series Modeling. *Pattern Recognition Letters*, 42:11–24, 2014.

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems 25*, pages 1106–1114, 2012.

[8] Q. V. Le, M'A. Ranzato, Monga R., Devin M., Chen K., Corrado G. S., Dean J., and Ng A. Y. Building High-Level Features using Large Scale Unsupervised Learning. In *IEEE 29th International Conference on Machine Learning*, pages 81–88, 2012.

[9] K. Noda, Y. Yamaguchi, K. Nakadai, H. G. Okuno, and T. Ogata. Audio-Visual Speech Recognition using Deep Learning. *Applied Intelligence*, 42(4):722–737, 2015.

[10] C. Anderson, I. Ebert-Uphoff, Y. Deng, and M. Ryan. Discovering Spatial and Temporal Patterns in Climate Data Using Deep Learning. In *5th International Workshop on Climate Informatics*, 2015.

[11] I. Lenz, H. Lee, and A. Saxena. Deep Learning for Detecting Robotic Grasps. *The International Journal of Robotics Research*, 34(4-5):705–724, 2015.

[12] Y. Bengio, P. Lamblin, D. Popovici, and H. Larochelle. Greedy Layer-Wise Training of Deep Networks. In *Advances in Neural Information Processing Systems 19*, pages 153–160, 2007.

[13] G. E. Hinton, S. Osindero, and YW. Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18(7):1527–1554, 2006.

[14] P. Vincent, H. Larochelle, Y. Bengio, and P. A. Manzagol. Extracting and Composing Robust Features with Denoising Autoencoders. In *25th International Conference on Machine Learning*, pages 1096–1103, 2008.

[15] H. Cecotti and A. Gräser. Convolutional Neural Networks for P300 Detection with Application to Brain-Computer Interfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):433–445, 2011.

[16] D. Erhan, Y. Bengio, A. Courville, and P. Vincent. Visualizing Higher-Layer Features of a Deep Network. Technical report, Université de Montréal, 2009.

[17] W. Samek, A. Binder, G. Montavon, S. Bach, and KR. Müller. Evaluating the Visualization of What a Deep Neural Network has Learned. *arXiv (preprint)*, 2015.

[18] J. Yosinski, J. Clune, A. Nguyen, T. Fuchs, and H. Lipson. Understanding Neural Networks Through Deep Visualization. In *Deep Learning Workshop, 32st International Conerence on Machine Learning*, 2015.

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

[20] G. E. Hinton. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Computation*, 14(8):1771–1800, 2002.

[21] K. H. Cho, T. Raiko, and A. Ilin. Gaussian-Bernoulli Deep Boltzmann Machine. In *The 2013 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7, 2013.

[22] N. Wang, J. Melchior, and L. Wiskott. An Analysis of Gaussian-Binary Restricted Boltzmann Machines for Natural Images. In *20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, pages 287–292, 2012.

[23] G. E. Hinton. A Practical Guide to Training Restricted Boltzmann Machines. Technical report, University of Toronto, 2010.

[24] Y. Tang. Deep Learning using Linear Support Vector Machines. In *Workshop on Representational Learning, 30th International Conference on Machine Learning*, 2013.

Author Biographies

Koki Kawasaki He recieved the bachelor of engineering degree in computer science from Nagoya University, Japan, in 2015. He is now studying for the master of engineering degree in computer science at Nagoya University. His main research topic is methods and applications of soft computing. Student member of IEEE, Institute of Electronics, Information and Communication Engineers.

Tomohiro Yoshikawa He recieved the Ph.D. degree in information electronics from Nagoya University, Japan, in 1997. He became a COE designed associate professor at graduate school of engineering, Nagoya University in 2005. Since 2006, he has become the associate professor at the same department. His main research topic is methods and applications of soft computing. Treasurer for Computational Intelligence society Japan chapter of IEEE. Member of IEEE, Japan Society for Evolutionary Computation, Japan Society of Artificial Intelligence, etc.

Takeshi Furuhashi He recieved the Ph.D. degree in electrical and electronic engineering from Nagoya University, Japan, in 1985. He became a professor at graduate school of engineering, Nagoya University in 2004. His main research topic is methods and applications of soft computing. Member of IEEE, International Fuzzy Systems Association, Japan Society of Kansei Engineering, etc.