Approach to Object Hardness Prediction by Rubber Ball Hardness Prediction Using Capsule Network

Shota Shindo[1], Takaaki Goto[2], Kensei Tsuchida[3]
[1] Next Solutions Corporation Limited, Kanagawa, Yokohama, Japan
[2,3]Department of Information Sciences and Arts, Toyo University, Saitama, Kawagoe, Japan

Article Info

Article history:

Received Aug 15, 2021
Revised Oct 26, 2021
Accepted Nov 10, 2021

Keywords:

Capsule network
Hardness
Regression analysis
Image process
Neural network

ABSTRACT

A hardness is often used as an index to compare similar objects such as fruits or wood. To measure an object's hardness, a hardness meter is required, and certain conditions must be met. The conditions are that the hardness meter is compatible with the object and must be close at hand. This research shows the possibility of measuring hardness without a hardness meter using a neural network. The method employs machine learning using a capsule network (CapsNet) of a neural network model. This research experimented using CapsNet with routing-by-agreement, CapsNet with expectation-maximization routing (EM routing) and the EM routing method with the addition of Tasks-Constrained Deep Convolutional Network (TCDCN). The four-layer CapsNet with EM routing implemented has achieved the state-of-the-art. Multi-layered CapsNet with EM routing was a very effective method for regression analysis as well. And, CapsNet has higher discriminative power using EM-routing than routing-by-agreement.

Corresponding Author:

Shota Shindo,
Next Solutions Corporation Limited,
1-13-19-704 Nishiku, Hiranuma, Yokohama, Kanagawa, Japan.
Email: sshindobusin@gmail.com

1. INTRODUCTION

A hardness is often used as an index to compare similar objects. For example, to compare the deliciousness of fruits, or to measure the aging of buildings. A hardness meter has certain conditions to measure. The condition is that the object and the corresponding hardness meter must be at hand. The corresponding hardness meter is needed chosen from among many types to measure accurately.

In order to overcome this condition, we considered a way to measure hardness from an image using a neural network (NN) of machine learning. The images of objects can be obtained via the Internet. Also, the hardness meter is limited to one type of a hardness meter by NN. There has been a lot of research on image recognition using NNs. There are two networks that can automatically extract features from images are the convolutional neural network (CNN) and the capsule network (CapsNet). Therefore, it supposed that the hardness predicted from the object image by the machine learning using the object image is as an input and a hardness value acquired from the object as a teacher data. However, it is difficult to predict the hardness of all objects in the world, so we will narrow down the range. We target rubber balls because our hardness meter is compatible with rubber balls. The purpose of this research is that creating a machine learning model which can predict the hardness of rubber balls from images using a neural network. We adopted CapsNet as the neural network because the results of Geoffrey et al. [1] and Youngjoo et al. [2] show that the performance is higher than CNN.

2. RELATED WORKS

There are many researches on approaches to perform regression analysis from images using CNN, including Spyros et al. [3], Shun et al. [4], and Jun et al. [5]. Also, there has been a research on regression analysis using CapsNet for predicting traffic speed [2]. The research results showed that CapsNet is more accurate than CNN.

This research develops the hardness detection method by CapsNet [6] and creates a model with higher than its performance. Also, the evaluation index has been changed to be suitable for the regression analysis. Calculating the hardness of an object from an image has been studied widely in many fields. In the field of food chemistry, there was a study to determine the texture of beef from images [7]. The authors analyzed manually the images, rather than an NN which can analyze features automatically. In the field of engineering, there is a research on measuring the hardness of aluminum alloys using deep neural networks (DNNs) [8, 9]. The hardness of the alloy was detected by inputting the hardness of the base atom of the alloy into the DNN. However, the hardness was not obtained from the alloy image. There is a research that used CNN to classify images into onomatopoeias of psychology (Shimoda et al. [10]). There are onomatopoeias representing the hardness, and onomatopoeias have succeeded in classifying images. However, this study only classified images and could not calculate the hardness value. In the field of psychophysics, there is research on the phenomenon that the hardness felt by the visual affects the tactile as a kind of the cross-modal effect as a kind of cross-modal effect [11, 12]. This research is expected that the image features that cause this phenomenon can be automatically extracted. In the fields of neuroscience and design, it has been found that humans need both tactile and visual information to predict the hardness of an object [13, 14, 15]. Therefore, if the information that enters the visual is changed while the hardness remains constant, the hardness felt by the tactile also changes. The image data for learning used in this research has not been changed so as to be different from the hardness measured by the hardness meter. Therefore, we expect discriminators to be deceived by making changes to the image such as humans.

No research existed on automatically extracting features from images to predict hardness, except for [6]. We provide a more performance method than the old research. Also, as a by-product, this engineering approach provides the field of psychology, psychophysics, neuroscience with the possibility that humans may use textural features to predict hardness.

3. RELATED NEURAL ARCHITECTURE

3.1. CNN

CNN is a type of NN that is often used for image recognition. In recent years, it also used in the sound recognition by imaging the features of sounds. An image recognition model using it won an award at the 2012 ILSVRC, an image recognition convention. CNN consists of convolution layers, pooling layers, full connecting layers and the input value is the image. The convolution layers use convolution filters to perform the convolution operation in front of the full connecting layers, similar to image processing. Weighting coefficients in the filters are updated by the gradient descent. These coefficients of the image filters can automatically extract image features. The pooling layers reduce the features from a convolution layer. There are three patterns of the reduction method: max-pooling, mean-pooling, and lp-pooling. These two layers (convolutional and pooling) are alternately calculated and finally become the input values of the full connecting layers. The full connecting layers perform like the ordinary DNN. CNN is a technique that is very often used in image processing and can create a powerful discriminator.

3.2. CapsNet with routing-by-agreement

CapsNet is a new NN based on CNN [1] (Figure 1). This network incorporates the unsupervised learning as part of the supervised learning. The pooling process in CNN is changed to routing-by-agreement (Figure 2) and the convolution process is changed from a scalar to a vector value. Routing-by-agreement can perform affine transformations for each input image. Therefore, CapsNet can cope with inputs from different angles of the images. From the experimental results using the rotated MNIST data by affine transformation, it was found that the accuracy of CapsNet is higher than CNN. Routing-by-agreement is looped an arbitrary number of times and learned by the unsupervised learning during detection, so it resembles a human thinking once and then making a decision. Regarding the MNIST data, the number of loops should be 5 or more as shown in Figure 3. In other experiments using two MNIST images were overlaid, CapsNet achieved higher accuracy than CNN. A major feature of CapsNet is that it learns the positional relationship of objects in an image. CNN searches only the characteristic parts of the images, so it detects regardless of the position of the characteristic parts. For example, if a picture of a face has its eyes and nose out of joint, CNN still recognizes it as a face. In this case, the detection is incorrect because the disjointed parts do not make it a proper face. By contrast, CapsNet recognizes the picture as a non face because it can detect objects by considering the positions of the individual parts. CapsNet is very powerful, but it has a weakness: it cannot recognize well an image which has a complicated background. This shortcoming is not limited to CapsNet. CNN also has a flaw: in another experiment [1], the accuracy of both networks was fairly low, but CNN was still slightly superior to CapsNet. However, CapsNet has the potential for improvement, such as the further deepening layers. It may become superior to CNN in the near future.

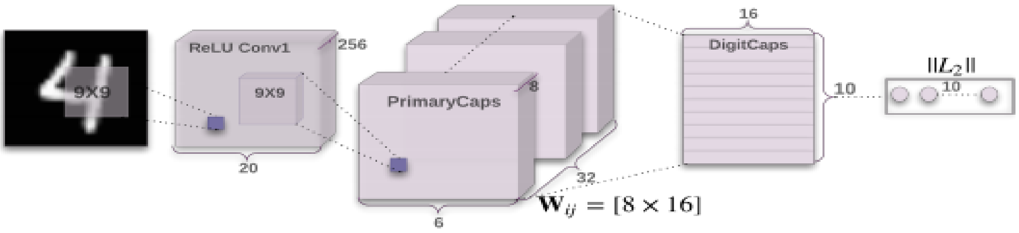


Figure 1. Structure of capsule network with routing-by-agreement (From the document *[1]*)

```
Parameters: $\hat{u}_{j|i}$ is an input vector made by capsule $i$ in layer $l$and capsule $j$ in layer $l + 1$.
            $r$ is an iteration count.
Procedure 1 Routing algorithm.
procedure Routing($\hat{u}_{j|i}$, $r$, $l$)
  for all capsule $i$ in layer $l$ and $j$ in layer $(l + 1)$: $b_{ij} \leftarrow 0$.
  for $r$ iterations do
    for all capsule $i$ in layer $l$: $\mathbf{c}_i \leftarrow \mathrm{softmax}(\mathbf{b}_i)$
    for all capsule $j$ in layer $(l + 1)$: $\mathbf{s}_j \leftarrow \sum_i c_{ij} \hat{u}_{j|i}$
    for all capsule $j$ in layer $(l + 1)$: $\mathbf{v}_j \leftarrow \mathrm{squash}(\mathbf{s}_j)$
    for all capsule $i$ in layer $l$ and capsule $j$ in layer $(l + 1)$: $\mathrm{b}_{ij} \leftarrow \mathrm{b}_{ij} + \hat{u}_{j|i} . \mathbf{v}_j$
return $\mathbf{v}_j$
```

Figure 2. Routing-by-agreement algorithm (From the document *[1]*)

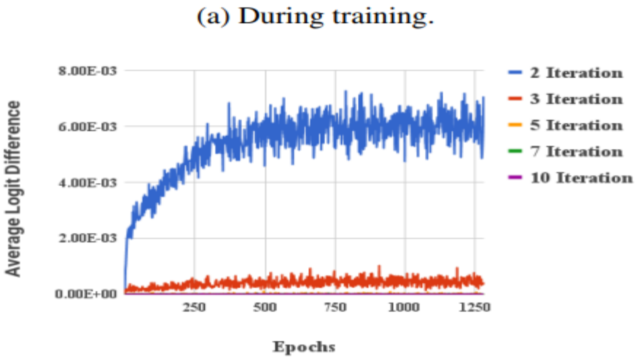


Figure 3. The number of routing algorithm loop (From the document *[1]*)

3.3. CapsNet with EM routing

CapsNet had a problem that how to make it multi-layered. Therefore, by changing routing-by-agreement to EM routing [16] which applies the EM algorithm used in unsupervised learning and speech recognition, multi-layering is possible. The machine learning method has also been changed from routing-by-agreement method. An output vector was treated as one value when routing-by-agreement was used. On the other hand, EM routing method uses two values called pose and activation, which are combined to form a capsule, and treated as one value. An example of multi-layered CapsNet and an EM routing algorithm are shown in Figure 4Figure *4* and Figure 5. The activation is treated as a numeric value for classification, and the pose is passed to an auxiliary error calculation model called reconstruction. The reconstruction is a sub loss function which is calculated separately from the main loss function such as the sigmoid function.

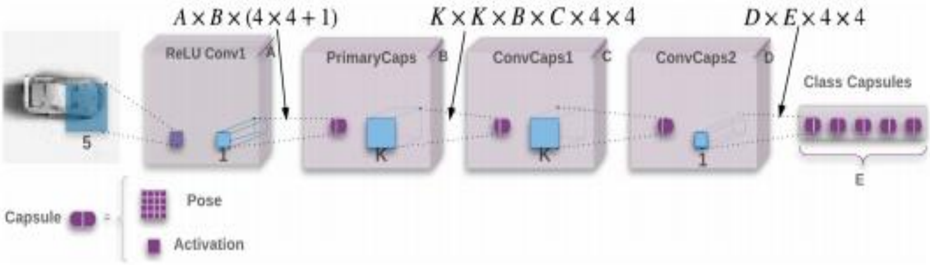

$A \times B \times (4 \times 4 + 1)$
$K \times K \times B \times C \times 4 \times 4$
$D \times E \times 4 \times 4$
ReLU Conv1
A
PrimaryCaps
B
ConvCaps1
C
ConvCaps2
D
Class Capsules
5
1
K
K
1
E
Capsule
=
Pose
Activation

Figure 4. Structure of capsule network with EM routing (From the document *[2]*)

Procedure 1 Routing algorithm returns **activation** and **pose** of the capsules in layer L+1 given the **activations** and **votes** of capsules in layer L. $V_{i,j}^{h}$ is the h^{th} dimension of the vote from capsule i with activation ai in layer L to capsule j in layer L+1. B_a, β_u are learned discriminatively and the inverse temperature λ increases at each iteration with a fixed schedule.

```
procedure EM ROUTING($\boldsymbol{\alpha}, V$)
  $\forall i \in \Omega_L \in \Omega_{L+1} : R_{ij} \leftarrow 1/|\Omega_{L+1}|$
  for $t$ iterations do
    $\forall j \in \Omega_{L+1}$ : M-STEP($\boldsymbol{\alpha}, R, V, j$)
    $\forall i \in \Omega_L$ : E-STEP($\mu, \sigma, \boldsymbol{\alpha}, V, i$)
  return $\boldsymbol{\alpha}, M$

procedure M-STEP($\boldsymbol{\alpha}, R, V, j$)
  $\forall i \in \Omega_L : R_{ij} \leftarrow R_{ij} * \boldsymbol{\alpha}_{\mathbf{i}}$
  $\forall h : \mu^h{}_j \leftarrow \frac{\Sigma \mathrm{i}\ Ri\ j\ V_{i,j}^{h}}{\Sigma \mathrm{i}\ Ri\ j}$
  $\forall h : (\sigma^h{}_j) \leftarrow \frac{\Sigma \mathrm{i}\ Ri\ j\ (V_{i,j}^{h} - \mu_j^h)2}{\Sigma \mathrm{i}\ Ri\ j}$
  $cost^h \leftarrow (\beta_u + log(\sigma^h{}_j))\ \Sigma_i R_{ij}$
  $\boldsymbol{\alpha}_{\mathbf{j}} \leftarrow logistic(\lambda(\beta_a - \Sigma_h\ cost^h))$

procedure E-STEP($\mu, \sigma, \boldsymbol{\alpha}, V, i$)
  $\forall j \in \Omega_{L+1} : \boldsymbol{P_j} \leftarrow \frac{1}{\sqrt{\prod_h^H 2\pi(\sigma_j^h)^2}} \exp\left(-\sum_h^H \frac{\left(V_{ij}^h - \mu_j^h\right)^2}{2(\sigma_j^h)^2}\right)$
  $\forall j \in \Omega_{L+1} : \boldsymbol{R_{i,j}} \leftarrow \frac{\boldsymbol{\alpha_j P_j}}{\sum_{k \in \Omega L+1} \boldsymbol{\alpha_k P_k}}$
```

Figure 5. EM routing algorithm (From the document *[2]*)

3.4. TCDCN

TCDCN has main loss function and sub loss function. The main loss function is the normal error between output and teacher datas. The sub loss function is errors between outputs and teacher datas added for training. For example, in face organ point detection, the error between the prediction of the face organ point and the teacher data is the error of the main loss function. On the other hand, the sub loss function is the error with additional teacher datas to classify the features of face images: wearing glasses, long face, looking sideways, etc. (Figure 6) The completed model uses only the output used by the main loss function. The regression analysis of face organ points is improved using this method [17, 18].

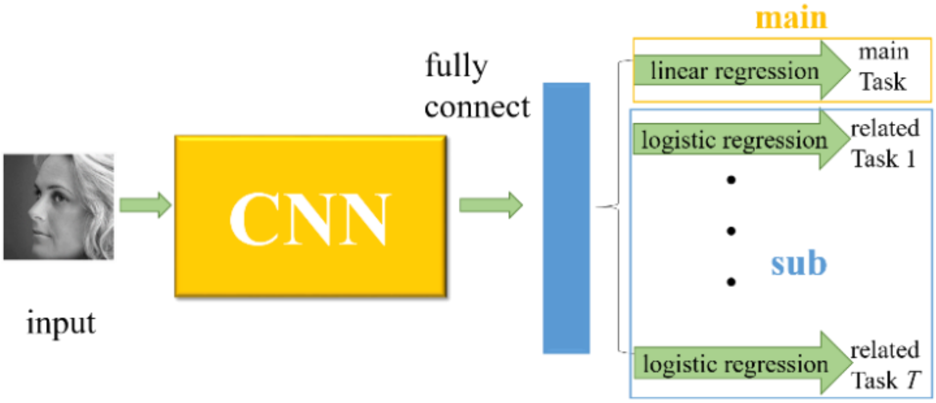


Figure 6. An example of TCDCN (From the document [18])

4. METHOD

4.1. Preparation

We collected rubber balls for experiments and take pictures. The rubber balls were photographed at a distance of 17cm from a camera. Then, the image of each ball was divided into 5 states. The hardness values of the balls were measured using a hardness meter, then the training dataset was created that corresponded to the images. The hardness meter was a needle-type durometer and was used according to the instruction manual (Figure 7). The power of pushing from above the hardness meter is 5kg. As shown in Figure 8 and Figure 9, states 1, 3, and 5 were for training data, while 2 and 4 were test data. The background is black with 0px because to evaluate with a simple background. The hardness values of the balls measured are shown in Table 1. The training datas were inflated by the data augmentation which moved 10px vertically and horizontally. There are 25 training datas and 15 test datas for the white ball. Also, There are 115 training datas and 55 test datas for the colored balls.

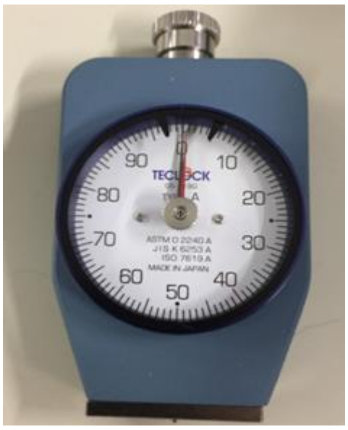


Figure 7. TECLOCK DUROMETER

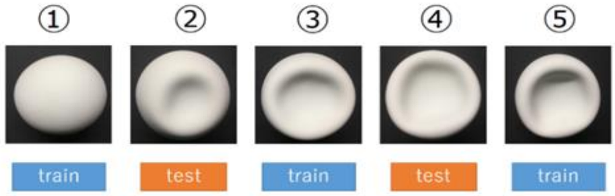


Figure 8. Sample images of a white rubber ball

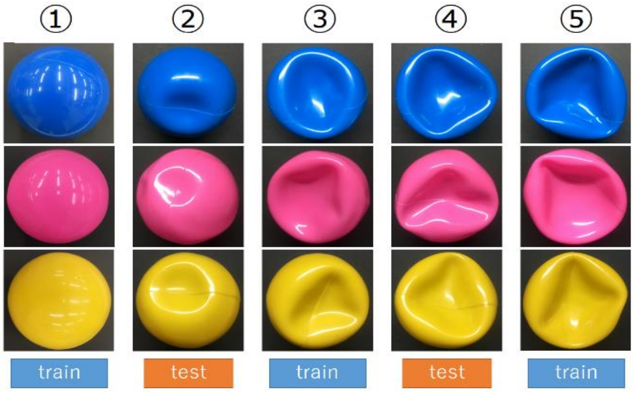


Figure 9. Sample images of a colored rubber ball

Table 1. Hardness value of each ball and state

	State 1	State 2	State 3	State 4	State 5
White ball	17	9	9	5	3
Blue ball	17	13	12	11	10
Pink ball	11	10	9	9	7
Yellow ball	14	14	12	11	11

4.2. Learning method and evaluation index

This research use multiple regression analysis using CapsNet that the input value is an image and the output is a hardness. The learning uses the ball states 1, 3 and 5 because to evaluate how predictable the hardness value of the balls not included in the learning (Figure 10). The multiple regression analysis changes the output vector, DigitCaps as shown in Figure 1, to one stage. The loss function is the squared error of the vector size and hardness value. The model performance uses Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), which are evaluation indexes. RMSE is defined as Eq. (1) and MAE is defined as Eq. (2).

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(t_i - y_i)^2} \quad (1)$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|t_i - y_i|^2 \quad (2)$$

where n is the number of test datas, i is the number of each test data, t is the teacher datas and y is the outputs in last layer.

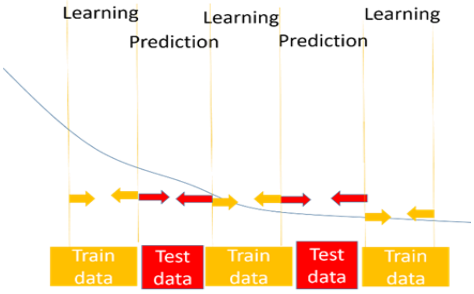


Figure 10. Multiple regression analysis of this research

4.3. Increased white ball's training datas

We increased white ball We increased white ball's training datas. A way to increase dataset is to draw a line on a white ball with a pen and take a picture again as different datas. The increased ball images are shown in Figure 11. The training datas for this white ball was added, and we learned by combining the training datas for white and colored balls. These hardness values and the number of states have been adjusted to match the white balls in Table 1. The number of additional data was 42 training datas and 24 test datas. The total datas for all white balls are 67 training data and 39 test data.

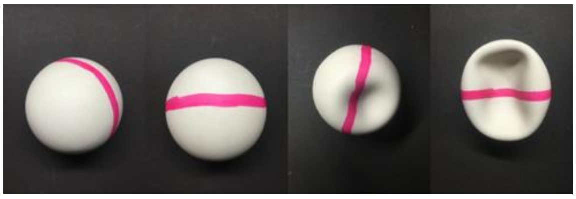


Figure 11. Added white ball images

5. EXPERIMENT

5.1. CapsNet with routing-by-agreement

Firstly, the input image was expanded to 256 images by a 2-dimensional convolution using a 9 × 9 filter. The activation function was ReLU. Secondly, a 3-dimensional convolution of 9 × 9 × 256 was performed to generate 256 images. This was divided into 8 images, and primary capsules were made as one capsule is 1 × 1 × 8 images. Finally, each capsule was multiplied by the weight W, and then executed by routing-by-agreement. The number of loops in routing-by-agreement is five times in this experiment. Adam optimizer is used to update the weights, and the number of updates is 100,000. In addition, the reconstruction is added as a sub loss function to the squared error.

We create a discriminator that has learned both white and colored balls. However, in this experience, we trained both separately for analysis, and also trained both grayscale images and RGB images. These results are evaluated using the accuracy to identify the cause of the large error. The threshold function was used to set an acceptable range for the accuracy. The threshold function is defined as:

$$threshold(x) = \begin{cases} 1, & x \le \sigma \\ 0, & otherwise \end{cases} \quad (3)$$

where x is the input datas and σ is the threshold.

The accuracy is calculated by entering errors into this function with the σ value being 1. The error exceeding this threshold is regarded as a large error in our analysis. The accuracy is defined as:

$$accuracy \ = \frac{1}{n}\sum_{i=1}^{n} threshold(|t_i - y_i|) \qquad (4)$$

where n is the number of test datas, i is the number of each test data, t is the teacher datas and y is the outputs in last layer.

5.2. Experiment using CapsNet with EM routing

We upgraded the CapsNet based on the experiment using CapsNet with routing-by-agreement, and experiment again. Training data and test data are RGB images of white and colored balls. The added white ball images are used. We multilayered CapsNet referring to the document [16], and incorporated grayscale images into the learning RGB images. The CapsNet has 4-layers which are the same as in Figure 4. When to use EM routing, the learning outputs two values which are a pose and an activation. However, how to handle these values in a regression is not described in the document [16]. Therefore, we conducted several experiments to find the most accurate method. As a result, a pose calculates the error with the training data and the activation calculates the error with 1. it turned out the highest accuracy. The loss function is defined as:

$$f(x) = \sqrt{(|y| - t)^2} \qquad (5)$$

where t is the teacher data and x is the activation or pose.

Originally, the activation is classified 1 if it belongs to a correct class and 0 if it does not belong. Therefore, in the case of the regression, we considered that the accuracy increased by the learning to always classify into one class. All losses in the program are defined as:

$$Loss = \beta \times active_{loss} + (pose_{loss} + \alpha \times reconloss) \qquad (6)$$

where β and α are learning rates, $active_{loss}$ is the loss between the activation and 1, *reconloss* is the reconstruction loss.

5.3. Experiment using CapsNet with EM routing attached TCDCN

We attached the TCDCN to With LastCaps. We called this model as CapsNet-EM TCDCN. The main loss function is the square error between the hardness value and the output as before. The sub loss function is the softmax cross entropy, and we classified trainig datas into two classes, white and color in the learning. The cross-entropy equation is defined as:

$$L \ = -\sum_{i=1}^{N} t_i \cdot \log\big(softmax(x_i)\big) \qquad (7)$$

where t is the teacher data, x is the output, and the number of outputs is N.

The pose and activation in the output are 1 set for the main loss function and 2 sets for the sub loss function, for a total of 3 sets. The pose of the sub loss function passed to the reconstruction to calculate an auxiliary error. The activation for the sub loss function was classified into two classes which are white and color. This neural network is shown in Figure 12**Error! Reference source not found.**.

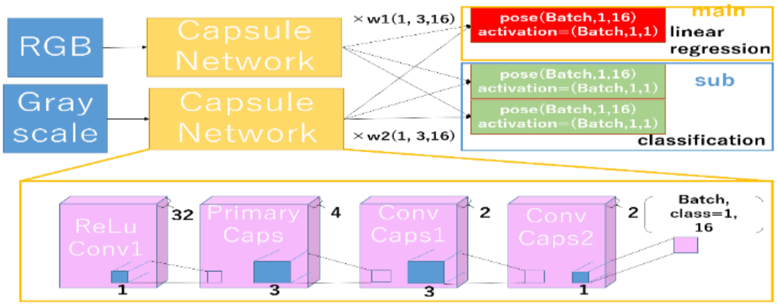


Figure 12. Structure of With LastCaps + TCDCN

6. RESULT AND DISCUSSION

6.1. Result of CapsNet with routing-by-agreement

First of all, images of the white balls, which were RGB and grayscale, were learned using CapsNet. The learning method is the online learning using 25 training datasets. Both loss functions converged to almost 0. As a result of the evaluation, the grayscale learning succeeded in detecting only state 2 and no state 4 was detected. Therefore, the accuracy rate was only 25%. The RGB learning failed all detections. Secondly, the colored ball images were learned with 115 training datasets in the same way as the white ball images. The RGB and grayscale loss function has converged to almost 0. As a result of the evaluation, the accuracy rate was 60% using grayscale images and 73% using RGB images. Both CapsNets model detected most states hardness. Thirdly, images that combined white and colored balls were learned using CapsNet. From the learning so far, there was a possibility that the white ball’s training dataset were insufficient because the accuracy rate of the white ball images was low, while the accuracy rate of the colored balls was high. The learning method is the batch learning with one batch is 70 using 140 training datasets. The loss function converged to almost 0 after nearly 60 thousand iterations. As a result of the evaluation using the test dataset of the combined white and colored balls, the accuracy rate was 62%. Although the accuracy rate was 0% when the learning using the white ball RGB images, the accuracy rate in this learning was 100% in state 2, and some state 4 hardness could be detected. However, the hardness of some of the colored balls was not detected and the accuracy was down. The detection results are shown in Table 2. Also, RMSE is 3.56 and the MAE is 1.48. Finaly, we added the additional white ball datas to the training data to learn the model in the same way as the third one. As a result, the accuracy of the white ball did not improve, and RMSE = 4.2 and MAE = 1.68.

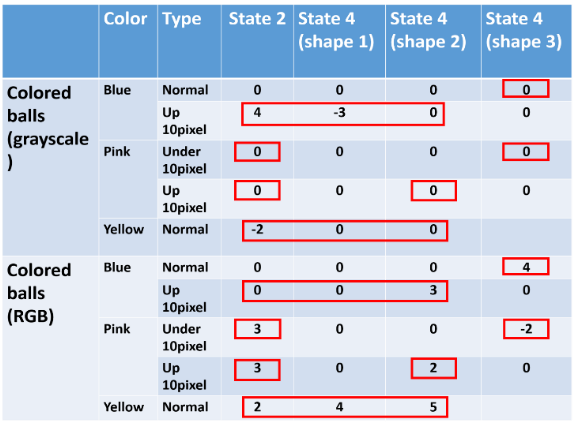
Table 2. Detection results

	State 2	State 4 (shape 1)	State 4 (shape 2)	State 4 (shape 3)	Average accuracy
White ball (Grayscale)	75%	0%	0%		25%
White ball (RGB)	0%	0%	0%		0%
Colored balls (Grayscale)	60%	60%	73%	70%	65%
Colored balls (RGB)	67%	80%	80%	60%	73%
White + Colored balls (RGB)	89%	42%	58%	60%	62%

6.2. Discussion of CapsNet with routing-by-agreement

The white ball’s training dataset was considered to be insufficient in the evaluation of the only white balls and the white + colored balls model. Also, the learning added only the colored balls’ training dataset was effective in overcoming the lack of a training dataset. However, the accuracy of colored balls RGB declined. Therefore, the addition of the white ball’s training dataset seemed to have a negative influence on predicting the hardness of the colored balls, which had an adequate training dataset. In short, the learning of white and colored ball images positively influenced hardness prediction of the white ball, which lacked a training dataset. However, it negatively influenced hardness prediction of the colored balls, which had an adequate training dataset. Accordingly, enlarging the same type of rubber ball training dataset is most desirable for a learning. Based on these results, we ran the experiment with increasing datasets of white balls, but the results did not improve. In later experiments, these increased images improve RMSE and MAE of the model. Therefore, it is considered that the discriminating power of this CapsNet is still low. Also, mixing the training dataset seemed difficult to learn, since the detection accuracy for the colored balls declined. On the other hand, the accuracy was high when the learning only the colored ball images. Considering this fact, there is a possibility that detection accuracy can be vastly improved using TCDCN (Figure 6). The method of TCDCN is to add outputs, and to teach model to classify colors.

Both grayscale and RGB training images of the colored balls resulted in high accuracy. But on closer examination, there were some images could be detected in grayscale images, but not in RGB images. Of course, there were images that showed the opposite pattern to these, as summarized in Figure 13. We compared the grayscale and color images, and the contrasting results are circled in red boxes. For example, the error in hardness prediction for a grayscale image type is normal and shape 4 is 0, but the error for the same type and shape in a color image is 4. Considering Figure 13, there is a possibility that image features which only grayscale or RGB images have. Especially, the possibility seems to very high because normal images that do not have data augmentation also follow this trend. Accordingly, the learning using both grayscale and RGB images as inputs supposed to improve accuracy because it is able to consider both types of an image.



	Color	Type	State 2	State 4 (shape 1)	State 4 (shape 2)	State 4 (shape 3)
Colored balls (grayscale)	Blue	Normal	0	0	0	0
		Up 10pixel	4	-3	0	0
	Pink	Under 10pixel	0	0	0	0
		Up 10pixel	0	0	0	0
	Yellow	Normal	-2	0	0	
Colored balls (RGB)	Blue	Normal	0	0	0	4
		Up 10pixel	0	0	3	0
	Pink	Under 10pixel	3	0	0	-2
		Up 10pixel	3	0	2	0
	Yellow	Normal	2	4	5	

Figure 13. Parts of detection of colored rubber ball

6.3. Result of CapsNet with EM routing

We trained a multilayering CapsNet using white ball, colored balls and added white ball images. The loss function is Eq. (6). In document [16], channels in CapsNet are A=32, B=32, C=32 and D=32 using A, B, C and D in Figure 4. However, the CapsNet having these channels did over learning, and its accuracy was 0%. Therefore, we did the machine learning which the channels greatly decreased (A=32, B=4, C=4 and D=4). If the learning is less than -log (0.01) = 4.6, all training data is detectable and successful. In routing-by-agreement, the iterator was 100,000 times, but the multilayering resulted in less than 4.6 at an early stage, so the iterator was greatly reduced to 1,500 times. This loss function is shown in Figure 14. The channels decreased to 44 (32+4+4+4) by multilayering against 544 (256+256+32).

From the result of the experiment using routing-by-agreement, we trained both grayscale and RGB images. The method is multiplying one variable and incorporating RGB calculation in front of the output layer. This method is called With LastCaps. The way of With LastCaps improved detection RMSE and MAE compared to the experiment using routing-by-agreement. In addition, we have succeeded in further improving them using added white ball images. RMSE and MAE were 2.8 and 1.4 when not using the additional datas. On the other hand, RMSE and MAE were 2.4 and 1.35 when using the additional datas.

6.4. Discussion of CapsNet with EM routing

This learning has achieved the state-of-the-art because he RMSE and MAE are smaller than the existing research [6]. This result suggests that multilayer CapsNet with EMrouting is effective for regression analysis. In addition, the method of training grayscale and color images together also seems to be effective. Moreover, learning with additional white ball datas, and RMSE and MAE improved. The method of adding white ball datas also supposed to be effective.

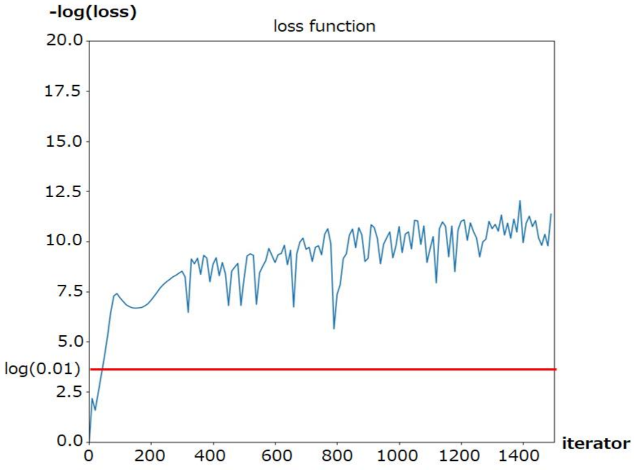


Figure 14. Loss function of 4-layers CapsNet

6.5. Result of CapsNet-EM TCDCN

We trained CapsNet-EM TCDCN on the same iterator as CapsNet with EM routing. The datasets are a combination of all white balls, colored balls and added white ball images. CapsNet-EM TCDCN is trained by classifying white balls and colored balls. Therefore, it is not possible to train a dataset of white balls or colored balls alone. Training results showed that the optimal number of channels for the model was A=32, B=8, C=4 and D=4. In the evaluation, only the output corresponding to the main loss function for learning regression is confirmed, and the result of the remaining two-class classification is discarded. This model has RMSE and MAE are 3.15 and 1.42.

6.6. Discussion of CapsNet-EM TCDCN

We expected that TCDCN would improve the results because the learning by mixing colored and white balls is difficult from the experiment of CapsNet with routing-by-agreement. However, the addition of TCDCN had no effect because RMSE and MAE were lower than CapsNet with EM routing. We suppose that CapsNet and TCDCN would not be a good match. Simply adding TCDCN may not be effective because CapsNet includes unsupervised learning unlike CNN.

6.7. Comparison of results

In this research, we trained and evaluated three type model using CapsNet. The first is CapsNet with routing-by-agreement (CapsNet-rba) using 3-layers CapsNet. The second is CapsNet with 4 layers, and grayscale and RGB images merged in the output layer (CapsNet-EM WithLastCaps). The third is CapsNet-EM TCDCN, which is a model that adds the function of TCDCN to the second model. RMSE and MAE values for these models are shown in Table 3**Error! Reference source not found.**. The evaluation index is a large value because almost hardness values are natural numbers. From the table, the highest performing model is CapsNet-EM WithLastCaps. This model can recognize hardness within an about 2.4 error on average because RMSE is 2.4. Furthermore, since the MAE is 1.35, which does not take into account large errors, it is considered that the model prediction errors are often 1 or 2.

Table 3. All models' evaluation results

Model name	RMSE	MAE
CapsNet-rba	3.58	1.48
CapsNet-EM WithLastCaps	2.4	1.35
CapsNet-EM TCDCN	3.15	1.42

7. CONCLUSION

The hardness prediction of the white ball and the colored balls using CapsNet succeeded in some state of each color. The hardness prediction of the white ball using black-and-white images succeeded the state 2 only due to the lack of training datas, and it using RGB images failed the all states. The hardness prediction of colored balls succeeded in both black-and-white images and RGB images. Finally, white + colored ball images were learned to compensate for the lack of white ball's training datas. This detection result was increased the accuracy rate when a white ball's test datas used, but decreased the accuracy rate when colored balls' test datas used. Based on this result, in this study, we increased white ball's training datas, multilayered CapsNet and did machine learning using both RGB and grayscale images. We added increased white ball images experimented again, but the model performance was down. In multilayering CapsNet using both RGB and grayscale images called CapsNet-EM WithLastCaps, this model combined in front of the output layer. RMSE and MAE were lower than the experiment using routing-by-agreement. In the detection by the model attached TCDCN option, RMSE and MAE were higher than With LastCaps. In conclusion, CapsNet-EM WithLastCaps is the best performance, RMSE and MAE were 2.4 and 1.35. This model has achieved the state-of-the-art. We showed the model structure in Figure 15.

This research's model has many errors with a hardness of 1 or 2 because less training datas. However, if we could get a large amount of training datas, our model has available that will provide higher performance due to the characteristics of supervised learning.

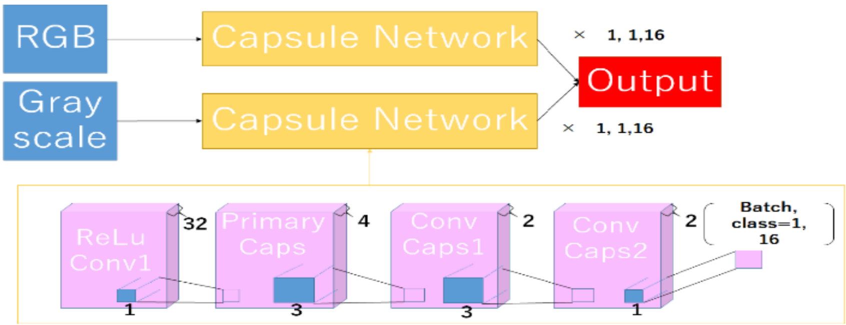


Figure 15. Structure of With LastCaps

REFERENCES

[1] Geoffrey E. Hinton, Sara Sabour, Nicholas Frosst, "Dynamic rouring between capsules," arXiv:1710.09829, 2017.

[2] Kim, Y., Wang, P., Zhu, Y. *et al.*, "A capsule network for traffic speed prediction in complex road networks", *International Journal of Applied Mathematics and Computer Science*, arXiv:1807.10603v2, 2018.

[3] Spyros Gidaris, Nikos Komodakis, "Object detection via a multi-region and semantic segmentation-aware cnn model", *International Conference of Computer Vision*, pp. 1134-1142, 2015.

[4] Shun Miao, Z. Jane Wang, Rui Liao, "A cnn regression approach for real-time 2d/3d registration", *IEEE Transactions on Medical Imaging*, vol. 5, no. 5, pp. 1352-1363, 2016.

[5] Jun Yuan, Bingbing Ni, Ashraf A.Kassim, "Half-cnn: A general framework for whole-image regression", arXiv:1412.6885, 2014.

[6] Shota Shindo, Takaaki Goto, Kensei Tsuchida, "Recognition of Object Hardness from Images Using a Capsule Network", *Proceedings of the 7th ACIS International Conference on Applied Computing and Information Technology*, 2019.

[7] J. Li, Jeryen Tan, P. Shatadal, "Classification of tough and tender beef by image texture analysis", *Meat Science*, vol. 4, no. 57, pp. 341-346, 2001.

[8] Adel M. Hassan, AbdallaAlrashdan, Mohammed T.Hayajneh, *et al.*, "Prediction of density, porosity and hardness in aluminum–copper-based composite materials using artificial neural network", *International Conference of Computer Vision*, vol. 2, no. 209, pp. 894-899, 2009.

[9] Bilal Zahran, "Using neural networks to predict the hardness of aluminum alloys", *Engineering, Technology &amp; Applied Science Research*, vol. 1, no. 5, pp. 757-759, 2015.

[10] Wataru Shimoda, Keiji Yanai, "Gathering and analyzing material images on the web with dcnn features", *Institute of Electronics, Information and Communication Engineers*, vol. 409, no. 114, pp. 67-72, 2015.

[11] Roland Harper, S. S. Stevens, "Subjective hardness of compliant materials", *Quarterly Journal of Experimental Psychology*, vol. 13, no. 16, pp. 204-215, 1964.

[12] S. S. Stevens, "On predicting exponents for cross-modality matches", *Perception & Psychophysics*, vol. 4, no. 6, pp. 251-256, 1969.

[13] Cristiano Cellini, Lukas Kaim, Knut Drewing, "Visual and haptic integration in the estimation of softness of deformable objects", *i-Perception*, vol. 8, no. 4, pp. 516-531, 2013.

[14] Martin Kuschel, Massimiliano Di Luca, Martin Buss, *et al.*, "Combination and integration in the perception of visual-haptic compliance information", *IEEE Transactions on Haptics*, vol. 4, no. 3, pp. 234-244, 2010.

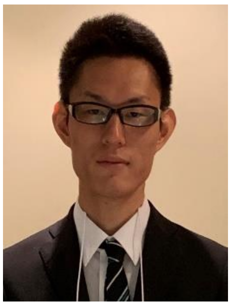
[15] Hideyoshi Yanagisawa, Kenji Takatsuji, "Effects of visual expectation on perceived tactile perception: An evaluation method of surface texture with expectation effect", *International Journal of Design*, vol. 1, no. 9, 2015.

[16] Geoffrey Hinton, Sara Sabour, Nicholas Frosst, "Matrix capsules with em routing", *Proceedings of ICLR*, 2018.

[17] Zhanpeng Zhang, Ping Luo, Chen Change Loy, *et al.*, "Learning deep representation for face alignment with auxiliary attributes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 5, no. 38, pp. 918-930, 2016.

[18] Zhanpeng Zhang, Ping Luo, Chen Change Loy, et al, "Faciallandmark detection by deep multi-task learning", *Proceedings of European Conference on Computer Vision*, 2014.

BIOGRAPHY OF AUTHORS



Shota Shindo received the master degree in computer science from the Department of Information Sciences and Arts , Toyo University, in 2020. His current research interests include machine learning, Computer learning theory, neural networks, Pattern recognition and artificial intelligence.

Takaaki Goto graduated with a Doctor of Engineering from Toyo University in 2009. From 2009 to 2015, He had been a Project Assistant Professor at the University of Electro-Communications in Japan. He was an Associate Professor at Ryutsu Keizai University in Japan from 2015 to 2019. Now he is an Associate Professor at Toyo University, Japan. His main research interests are applications of graph grammars, visual languages, and software development environments. He is a member of ACM, IEEE, ACIS, ISCA, IEICE, and IPSJ.

Kensei Tsuchida received M.S. and D.S. degrees in mathematics from Waseda University in 1984 and 1994, respectively. He was a member of the Software Engineering Development Laboratory, NEC Corporation in 1984–1990. From 1990 to 1992, he was a Research Associate of the Department of Industrial Engineering and Management at Kanagawa University. In 1992 he joined Toyo University, where he was an Instructor until 1995 and an associate professor from 1995 to 2002, and a Professor from 2002 to 2009 at the Department of Information and Computer Sciences, and since 2009 he has been a Professor of Faculty of Information Sciences and Arts. He was a Visiting Associate Professor of the Department of Computer Science at Oregon State University from 1997 to 1998. His research interests include software visualization, human interface, graph languages, and graph algorithms. He is a member of IPSJ, IEICE Japan and IEEE Computer Society.