Pre-training of Pneumonia Classifier for Chest CT images using Fractal Database

Yuken Yoshioka[1], Daichi Ikefuji[2], Tomokazu Funatsu[1], Takashi Nagaoka[3], Takenori Kozuka[4], Mitsutaka Nemoto[5], Takahiro Yamada[6], Yuichi Kimura[7,8], Kazunari Ishii[4,6], and Hitoshi Habe[7,8]

[1] Graduate School of Science and Engineering, Kindai University, Japan
[2] Department of Informatics, Faculty of Science and Engineering, Kindai University, Japan
[3] Department of Computational Systems Biology, Faculty of Biology-Oriented Science and Technology, Kindai University, Japan
[4] Department of Radiology, Faculty of Medicine, Kindai University, Japan
[5] Department of Biomedical Engineering, Faculty of Biology-Oriented Science and Technology, Kindai University, Japan
[6] Institute of Advanced Clinical Medicine, Kindai University Hospital, Japan
[7] Department of Informatics, Faculty of Informatics, Kindai University, Japan
[8] Cyber Informatics Research Institute, Kindai University, Japan

Abstract. Recently, the number of images for pre-training of deep learning models has been increasing, and large-scale data sets contain inappropriate images such as ethically inappropriate images, copyright infringement, and labeling errors. A method to solve these is by using a fractal database that generates images by mathematical formulas without using natural images. Our goal is to show that the classification accuracy obtained by pre-training with fractal images is comparable to natural images. In the experiments, we compare the performance on the tasks to classify CT images of COVID-19 pneumonia and regular pneumonia.

Keywords: Image classification · Fractal Image · Pre-training · CNN · CT image

1 Introduction

Imaging diagnosis by doctors is essential for the detection of disease. However, if image diagnosis is performed by doctors, even doctors sometimes fail to detect lesions from large amounts of data. This would cause a problem of delays in treatment. Image recognition using deep learning would enable us to find diseases early.

When we perform image recognition using deep learning(DL), first, we perform pre-training using a large, public dataset, and then the DL model is updated by fine-tuning using the data of the application field. In recent years, the required number of datasets for pre-training has been increasing to improve the accuracy of DL, and the creation cost of datasets, including image collection and annotation, has also been increasing. Furthermore, there are unignorable problems in the public dataset, such as labeling errors[1].

To overcome such problems, Kataoka et al. propose to generate a large amount of training data based on a mathematical model and to use the data for pre-training of DL[2]. Kataoka et al.[2] have shown that pre-training using a formula-driven database for natural image classification tasks yields results that are comparable to those of conventional large-scale natural image databases such as ImageNet. In this study, we use a fractal image database, a mathematical model-based database. And we demonstrate its effectiveness for CT image classification tasks. In the following discussion, we call the fractal image database Fractal Database(FDB).

2 Generating a Fractal Database

In this section, we describe a method for generating fractal Database[2].

2.1 Generating Fractal Images

IFS (Iterated Function System) is a model for generating the point set $X = \{x_1, x_2, \ldots, x_K\}$. IFS is defined by a set of transformations $w_i : X \to X$ and corresponding probabilities p_i in the complete metric space X (Eq. (1)). where N represents the number of pairs (w_i, p_i).

$$IFS = \{X; w_1, w_2, \ldots, w_N; p_1, p_2, \ldots, p_N\}. \tag{1}$$

Using the IFS, a fractal $S = \{x_t\}_{t=0}^{\infty} \in X$ is constructed by the random iteration algorithm.

The transformation w is defined as:

$$w_i(x; \theta_i) = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix} x + \begin{bmatrix} e_i \\ f_i \end{bmatrix}. \tag{2}$$

$\theta_i = (a_i, b_i, c_i, d_i, e_i, f_i)$ is 6 parameters for rotation and shifting. It generates a point set to depict a fractal image in the two-dimensional Euclidean space. p_i is the probability for selecting the transformation w, and is calculated as follows:

$$p_i = \frac{|detA_i|}{\sum_{i=1}^{N} |detA_i|}, \qquad A_i = \begin{bmatrix} a_i & b_i \\ c_i & d_i \end{bmatrix}. \tag{3}$$

The following procedure obtains the point set X that depicts a fractal image.

I Determine the number N of pairs (w_i, p_i) from a discrete uniform distribution on $[2, 3, ..., 8]$.

II The parameters a_i, b_i, $\ldots$, f_i in the transformation w_i in Eq. (2) are randomly selected from the continuous uniform distribution on $[-1, 1]$ and the probability p_i is determined by Eq. (3).

III Set the initial value to $x_0 = (0, 0)^T$. Choose the transformation w_i from $w_1, \ldots, w_N$ with according to the probability p_i. Apply it to the position x_{t-1} to obtain the new position x_t.
IV By repeating (III) K times, we obtain a point set $X = \{x_1, x_2, \ldots, x_K\}$.

When we draw a fractal image from the point set, we randomly choose a value of 0 to 127. And draw the values at the 3×3 region centered on the obtained point set $X = \{x_1, x_2, \ldots, x_K\}$. Get the maximum and minimum x and y coordinates of the point set X to normalize the size into a pre-defined image size. Finally, calculate the pixel filling rate. If it is above a certain threshold value, it is used as an image for pre-training.

We assign the same category label for images generated by the same IFS. IFS is characterized by a set of parameters and their corresponding probabilities, expressed as $\boldsymbol{\theta} = \{(w_i, p_i)\}_{i=1}^{N}$. When we create a data set of n classes, we repeat steps (I)-(IV) at n times. In this research, we use FDB having 1000 classes. The procedure for generating more data for each class is described in the next section.

2.2 Data Augmentation

In section 2.1, we generated one image per class. These classes are related by fractal parameters $a_i, b_i, \ldots, f_i$. Because this is insufficient for training, it is necessary to augment the data for each class, as is often performed in the standard training procedure. We apply the two types of augmentation methods in this paper shown in Figure 1. The first one is based on the original paper[2]. The other one is the image-based augmentation conventionally performed in the standard training process[3].

Formula-based Augmentation As used in [2], we increase the data by the three types of augmentation methods as follows.

I Changing the parameter set of IFS.
II Rotate generated fractal images.
III Changing 3×3 patch patterns for drawing fractal images.

We carry out (I) 25 times, (II) 4 times, and (III) 10 times. Finally, we generate the database containing 1000 images per class. We call this database FDB1k-1k.

Image-based Augumentation The data augmentation described in Section 2.2 draws fractal images for each parameter of IFS. This process is quite time-consuming. Instead of this, we apply simple data augmentation for training[3]. As is the standard data augmentation process, we apply affine transformations and color transformations using random numbers to each fractal image using the RandomAffine function and ColorJitter function in PyTorch libraries. This would enables us to obtain sufficient number of training data for pre-training as is the previous section. We call the augmented data One-instance Fractal Data Base (OFDB). More specifically, we use OFDB1k and OFDB10k data sets which contains 1000 classes and 10000 classes respectively.

Fig. 1: FDB and OFDB data sets

3 Training Pneumonia Discriminator

This Section describes the precedure for training pneumonia discriminator using FDB1k-1k, OFDB1k, and OFDB10k.

3.1 Pre-training

For comparing the pre-training results of fractal data sets, we use ImageNet Version 2 (IN_V2). IN_V2 is a pre-trained model of 1000 class classification using ResNet50 as the backbone network. Table 1 shows the pre-training conditions. As mentioned is Section 2.2, FDB1k-1k need to generate point set for each images, i.e. 1000 sets for each class. On the other hand, OFDB uses PyTorch function for data augmentation, which realizes efficient computation. Therefore, it has the advantage of short learning time.

Table 1: Pre-training conditions

	FDB1k-1k	OFDB1k	OFDB10k
Hours	168	8	27
Epochs	90	9000	900
Class	1000		10000
Network	ResNet-50		
Batch_size	64		

3.2 Fine-tuning

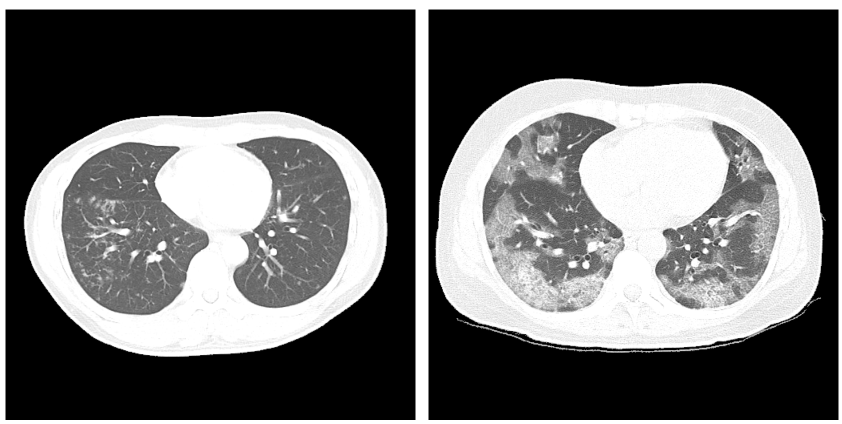
Next, we perform fine-tuning to adapt the pre-trained models to the pneumonia data. In this paper, we aim the distinguishes between two classes of regular pneumonia and COVID-19 pneumonia. We set the fine-tuning conditions: the epochs number of 90, the batch size of 64, and the learning rate of 0.01.

4 Experiment

We conduct the experiment to examine the effectiveness of the fractal database.

4.1 Experimental Setup

Figure 2 show the two examples of CT images for experiments. (a) is normal pneumonia and (b) is COVID-19 pneumonia. These images were provided by Kindai University Hospital under the permission of the Ethics Committee, Kindai University Faculty of Medicine. Using a data set containing these images, we perform a two-class classification of COVID-19 pneumonia or other. Because COVID-19 pneumonia has frosted-glass shadows in CT images, those features would be an essential cue for classification. 10-fold cross-validation is used for verification. The number of images used is 7092 for regular pneumonia and 6621 for COVID-19 pneumonia.



(a) Normal pneumonia (b) COVID-19 pneumonia

Fig. 2: Example of normal pneumonia and COVID-19 pneumonia

4.2 Results

Table 1 and Figure 3 show the results of 10-fold cross-validation. In Table 1, we show the average of the highest accuracy and recall rate for the test data among 90 trials. The accuracy of IN_V2 was the highest at 90.84%, followed by OFDB1k at 86.43%, OFDB10k at 85.27%, and FDB1k-1k at 85.53%. However, we have to note that decreasing the false-negative, i.e., missing COVID-19 cases, is crucial for diagnosis. From this point of view, the OFDB1k and OFDB10k yield slightly better results than IN_V2. Figure 3 shows the box plot of accuracy and recall during 10-fold cross-validation. This also shows that there is almost no difference between the recall values for OFDB1k, OFDB10k, and IN_V2, while the accuracy of IN_V2 is higher than others.

Table 2: 10-fold cross-validation(%)

	FDB1k-1k	OFDB1k	OFDB10k	IN_V2
Accracy	85.53	86.43	85.27	90.84
Recall	91.54	95.51	95.26	95.48

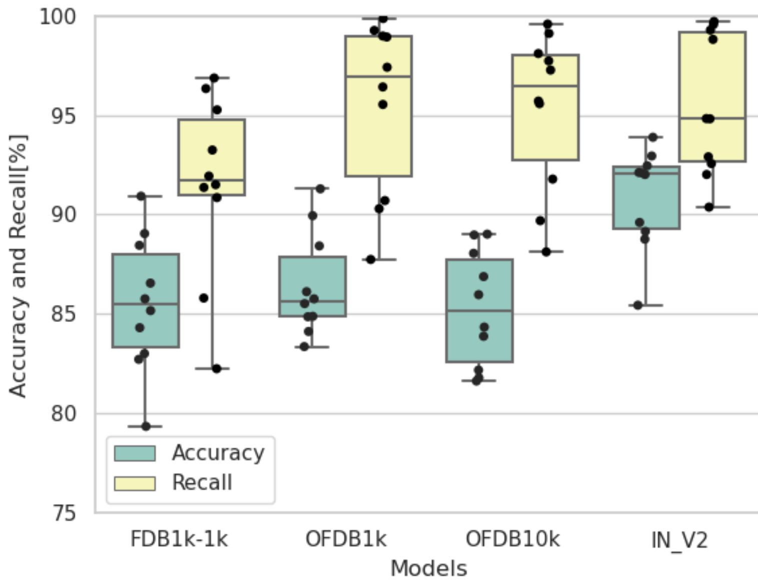
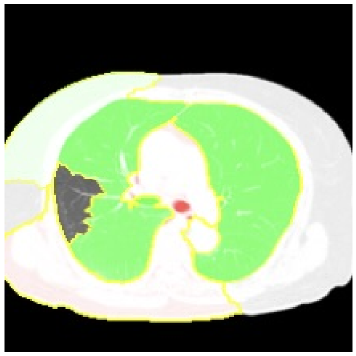


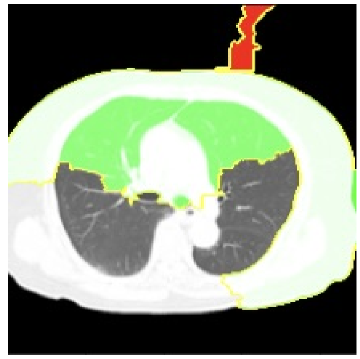
Fig. 3: Box plot of accuracy and recall during 10-fold cross-validation

4.3 Visual Explanations for Decisions

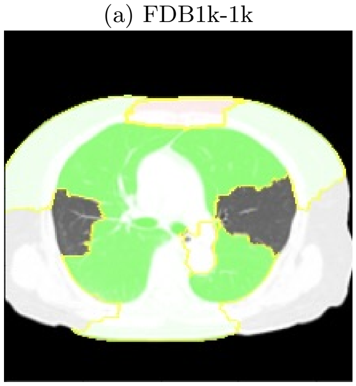
Figure 4 shows the visualization of the reasons behind predictions by LIME[5]. This shows the parts that significantly contribute to the classification. While the green color represents the part that has a higher predicted probability of the correct label, the red color shows a lower probability. From this figures, we can see that the models trained by the fractal database, i.e., FDB and OFDB, focus on the larger regions in the lung area.



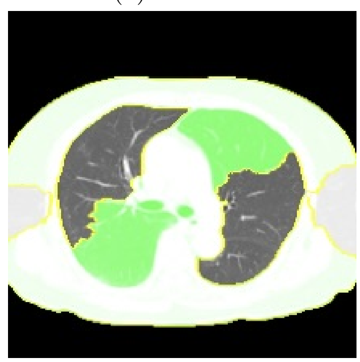
(a) FDB1k-1k



(b) OFDB1k



(c) OFDB10k



(d) IN_V2

Fig. 4: Visualization of attention area by LIME

5 Conclusion

Although the accuracy of classification using FDB and OFDB are lower than IN_V2, the recall values of OFDB are almost the same as the IN_V2 model. OFDB has the advantage that this has less computational costs. It needs only one single image for each class. Data augmentation can be efficiently conducted by PyTorch libraries.

From section 4.3, FDB and OFDB focus more correctly in the lungs. This implies that the fractal-based database has the potential to achieve higher performance if we have sufficient data for fine-tuning because the model trained by FDB and OFDB focuses on appropriate regions.

In future work, we will investigate the appropriate model to generate a database for pre-training. In the current implementation, we use the model same as the original paper. It is expected that higher performance can be obtained by exploring database generation models that are suitable for the data in the application domain. Additionally, there are still the room for optimizing the hyper-parameters for pre-training, which helps to obtain more reliable models.

This work was partly supported by JSPS KAKENHI Grant Number JP21H05302 and All-Kindai University support project against COVID-19 provided by Kindai University.

References

1. Lucas Beyer, Olivier J. Hénaff, Alexander Kolesnikov, Xiaohua Zhai, and Aäron van den Oord. 2020. "Are We Done with ImageNet?" arXiv [cs.CV]. arXiv. http://arxiv.org/abs/2006.07159.
2. Hirokatsu Kataoka, Kazushige Okayasu, Asato Matsumoto, Eisuke Yamagata, Ryosuke Yamada, Nakamasa Inoue, Akio Nakamura, and Yutaka Satoh. 2022. "Pre-Training Without Natural Images." International Journal of Computer Vision 130 (4): 990–1007.
3. Ryo Nakamura, Ryu Tadokoro, Hirokatsu Kataoka, Can visual features be learned with only one generated image per category?, MIRU2022(in Japanese).
4. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. "Deep Residual Learning for Image Recognition." In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 0:770–78.
5. Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. 2016. "'Why Should I Trust You?': Explaining the Predictions of Any Classifier." In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1135–44. KDD '16. New York, NY, USA: Association for Computing Machinery.