Deep Learning Model for Predicting Preference of Space by Estimating the Depth Information of Space using Omnidirectional Images

Hina Kinugawa[1], Atsushi Takizawa[2]
[1,2]Osaka City University
[1]kinugawa.hina@gmail.com [2]takizawa@osaka-cu.ac.jp

In this study, we developed a method for generating omnidirectional depth images from corresponding omnidirectional RGB images of streetscapes by learning each pair of omnidirectional RGB and depth images created by computer graphics using pix2pix. Then, the models trained with different series of images shot under different site and weather conditions were applied to Google street view images to generate depth images. The validity of the generated depth images was then evaluated visually. In addition, we conducted experiments to evaluate Google street view images using multiple participants. We constructed a model that estimates the evaluation value of these images with and without the depth images using the learning-to-rank method with deep convolutional neural network. The results demonstrate the extent to which the generalization performance of the streetscape evaluation model changes depending on the presence or absence of depth images.

Keywords: *Omnidirectional image, depth image, Unity, Google street view, pix2pix, RankNet*

INTRODUCTION

In the fields of architecture and urban planning, analysis is commonly performed to evaluate the impression of space. For instance, the sky ratio and green vision rate are commonly used; these metrics indicate the amount of visible sky and amount of green, respectively. To conduct such research, a comprehensive image database linked to a map is necessary. Because large-scale image data such as Google Street View (GSV) is available, image analysis is becoming feasible. Generally, when evaluating spaces using images, it is necessary to extract meaningful and calculable features from the images. However, features that affect the impression of space are not always exhaustive and can thus be defined explicitly.

In recent years, there have been rapid advances in deep convolutional neural networks (DCNNs). These networks automatically extract features from images and are used for classification, regression, and other tasks. DCNNs have become a fundamental technology in the current field of artificial intelligence. For example, Liu et al. (2017) and Law et al. (2017) conducted studies modeling urban landscape evaluation and classification using DCNNs with a large quantity of GSV images. Their studies used RGB images with a normal angle of view. RGB im-

ages are deemed suitable for capturing the color and textural features of space. However, geometric features, such as openness and size, are also important for evaluating space. Although these features have been conventionally studied using a space analysis method called Space Syntax (Hiller and Hanson, 1989), it is difficult to directly obtain geometric features using current image analysis methods.

Based on the above, Takizawa and Furuta (2017) shot a large number of omnidirectional RGB images and depth images in a virtual urban space constructed with the game engine Unity[1] (see Figure 1). Then, using these images as input data, they constructed an evaluation model to estimate the results of another evaluation experiment of a virtual urban landscape with a DCNN. Their results revealed that adding depth images to ordinal RGB images made it possible to construct a DCNN with higher precision and easier interpretation. It should be noted that the omnidirectional depth image is a three-dimensional version of the so-called isovist (Benedikt, 1979; see Figure 2). Whereas previous approaches involving Space Syntax used a geometrical approach, Takizawa and Furuta proposed a new spatial analysis method based on image analysis using a DCNN and omnidirectional images.

Although accurate depth images can be generated using computer graphics (CG), obtaining depth images of an actual urban space is not simple. For example, even with increasingly popular three-dimensional laser scanners and photogrammetry, it remains difficult to obtain low-cost and comprehensive city-scale depth images.

In this study, we developed a method for generating omnidirectional depth images from corresponding omnidirectional RGB images of streetscapes by learning each pair of omnidirectional RGB and depth images created by CG using pix2pix (Isola et al., 2017), a general-purpose image conversion method based on deep learning. Then, the models trained with different series of images shot under different site and weather conditions were applied to GSV images to generate depth images. The validity of the generated depth images was then evaluated visually. In addition, we conducted experiments to evaluate GSV streetscape images using multiple participants. We constructed a model that estimates the evaluation value of these images with and without the depth images using the learning-to-rank method with DCNN. The results demonstrate the extent to which the generalization performance of the streetscape evaluation model changes depending on the presence or absence of depth images. Finally, we evaluated the efficiency of the proposed method.

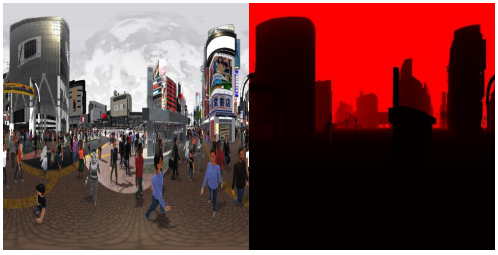


Figure 1
Omnidirectional RGB (left) and its depth (right) images of a streetscape in a computer graphics model

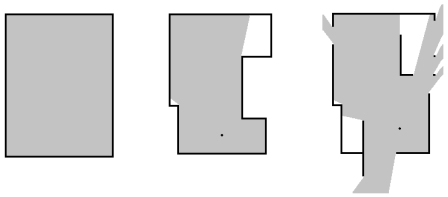


Figure 2
Example of two-dimensional isovists

The SYNTHIA dataset (Ros et al., 2016) is a pioneering attempt to generate artificial images of a large number of urban landscapes, including omnidirectional depth images, using CG for training a deep learning model. However, there are several differences between their study and the present study. Specifically, our study targets Japanese streetscapes, and the shooting height of an image is set to the height of the GSV on-board camera. In addition, only the depth information of the built and natural environment without cars and pedestrians is required. We can thus construct a unique dataset of urban spaces and use it for learning.

PROPOSED METHOD 1: GENERATING DEPTH IMAGES

The following includes step-by-step explanations of our research method.

Building three-dimensional urban space models

As in previous studies, the game engine Unity was used to develop three-dimensional models of the target city. In this study, we used two three-dimensional urban models that are realistic and faithful to the actual Japanese urban space sold in CG markets. They include the Shibuya model[2], which simulates the central zone of the Shibuya area, and a local city model[3], which simulates a suburban district in Japan (see Figures 3 and 4).

Figure 3
Shibuya model

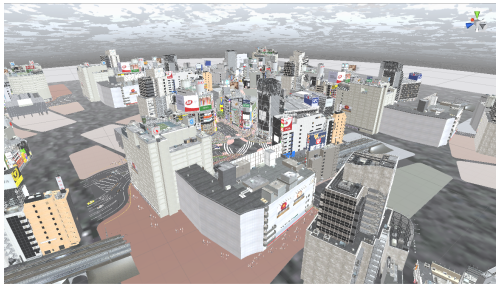


Figure 4
Local city model

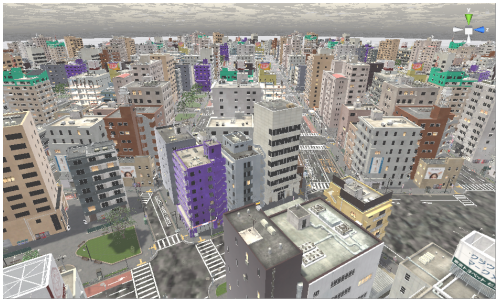


Figure 5
Shooting points in local city model

To apply the trained model with GSV images, we added pedestrian and car models to the street and attempted to reproduce an actual urban space. Because the area of the original three-dimensional models was not sufficiently wide to shoot depth images, the original model was copied, and the street area was expanded multiple times. As mentioned in the Introduction, the depth images used as training data excluded objects that may have served as obstacles in the space, such as pedestrians and cars, in order to acquire only the fixed spatial information of the built and natural environment. In addition, because actual photos of streetscapes vary greatly depending on weather, season, and time, even for a single location, we used the Tenkoku Dynamic Sky[4] weather asset of Unity to modify the environmental conditions. Specifically, because sky conditions greatly affect the generation of depth images, two sky conditions-blue and cloudy skies-were considered.

Shooting omnidirectional images

For each of the two city models provided, the camera height was set to 2.05 m, which is the height of the GSV shooting car's camera in Japan. Next, as illustrated in Figure 5, a planar image of a three-dimensional model was input to the GIS, and a large number of shooting locations were randomly set on the road, with 550 points for each model. The coordinates of the observation points were imported into Unity, and omnidirectional RGB and depth images were taken at each point. For shooting omnidirectional images, we used a discontinued camera asset of Unity called Spherical Image Cam. The number of data points was increased by shooting 18 omnidirectional images by rotating the camera 20 degrees at a single point.

As a result, we obtained 18 * 550 = 9,900 images for each three-dimensional model. The images shot from inside a building were excluded and used for learning. The size of each image output was 256 × 256.

Implementing datasets for learning

Using pix2pix, we trained the model to generate depth images from omnidirectional RGB images. Pix2pix is an image generation model based on deep learning; it is a technology that generates images by learning the relationship between pairs of images. Figure 6 presents an example of depth image generation using pix2pix. Figure 6 (left) presents the input RGB image, (center) presents the input real depth image, and (right) presents the generated depth image by pix2pix.

We prepared 3 types of spatial datasets for learning: Shibuya, a local city, and a mix of the two. We also prepared three sky conditions: blue sky, cloudy sky, and a mix of the two. By combining the datasets, we generated nine training datasets, as listed in Table 1.

Table 1. Nine models used for training

	Sky condition		
Spatial model	a. Blue sky	b. Cloudy sky	c. Mix of a and b
1. Shibuya	M1a	M1b	M1c
2. Local city	M2a	M2b	M2c
3. Mix of 1 and 2	M3a	M3b	M3c

We divided the images into training and validation datasets. The ratio of training data to validation data was set to 3:1 for each dataset. Thus, we used a total of 10,044 images for M1a and M1b, 3,925 images for M2a and M2b and 13,929 images for M3a and M3b models as the training dataset. The data set of M1c is the sum of the images of M1a and M2b. M2c and M3c also have the same configuration.

RESULTS OF TRAINING AND VERIFICATION OF METHOD 1

Here, we describe the results of the training and verification of Method 1.

Training pix2pix models

We used the PyTorch[5] implementation[6] of pix2pix for training and verification. We changed the number of epochs to 400; however, the other hyperparameters of pix2pix were left as default. We performed learning for each model. Pix2pix consists of a generator that generates an image, and a discriminator that discriminates whether the image is real or fake. In this study, we confirmed the accuracy of the model from the convergence of the loss function of the discriminator (D_fake). Figure 7 presents an example of the convergence of the loss function when M2b was trained up to 400 epochs. Although some fluctuation was observed, the value of the loss function decreased with the learning progress.

Verification of trained pix2pix model by GSV

We input the omnidirectional GSV images to the trained pix2pix model and obtained an estimated depth image. Then, we visually evaluated how realistic the generated depth image was. The GSV images were sampled from urban areas in the Osaka Prefecture. Examples of the results are presented in Figure 8. Of the nine models listed in Table 1, the M2b model exhibited stable and superior performance throughout the whole image. Other models, such as MXa (X=1,2,3), produced inaccurate distances in the sky.

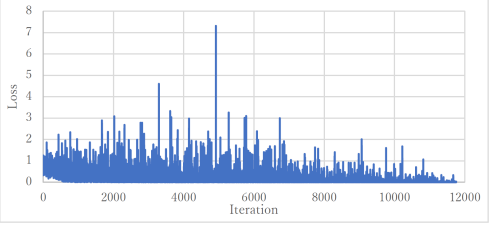


Figure 6
Example of generation of depth images by pix2pix. (Left) Input RGB image (center) Real depth image of (left) (right) Generated depth image

Table 1
Nine models used for training

Figure 7
Example of convergence process of loss function (D_fake).

Figure 8
Estimation results of GSV depth image.

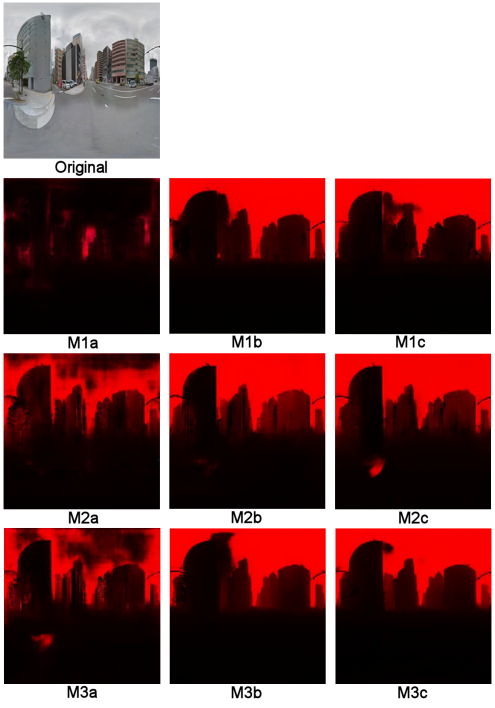


Figure 9
Examples from GSV impression evaluation experiment in Osaka. (Left) Good (3.4) (center) Moderate (2.2) (right) Poor (1.0)

PROPOSED METHOD 2: PREDICTING THE EVALUATION OF GSV IMAGES USING ESTIMATED DEPTH IMAGES

Using the estimated depth images, we estimated the evaluation of GSV images. An outline of this process is presented below.

Impression evaluation experiment

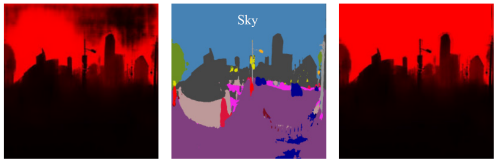
A GSV image was projected via the Oculus Rift, and an impression evaluation experiment was conducted to generate learning-to-rank data. Fifty GSV images were sampled from a local area, and the same number of images was sampled from an urban area. Both areas were located in the Osaka Prefecture, and a total of 100 images were used for the experiment. The streetscape of the collected GSV images was evaluated in four stages: good = 4, moderately good = 3, moderately bad = 2, and poor = 1. Ten university students majoring in architecture were selected as participants, and each participant evaluated 50 images considering their fatigue of evaluation. Each image was randomly selected and presented to exactly five participants.

The average evaluation value from five participants was used as the evaluation value for each image as an objective variable of the learning-to-rank model described later. Figure 9 presents an example of images of places with high, average, and low evaluation values provided by all participants.

Generation and filtering of RGBD images from GSV

Using the trained depth image generation models described above, estimated depth images were generated for each of the 100 GSV images used in the evaluation experiment. Depth estimation was performed with the M2b model, as it produced superior results by visual observation. However, because there was incorrect estimation of the sky area, as seen in Figure 10 (left), the sky area was extracted by semantic segmentation (center), and the value of the corresponding pixel was changed to red (right).

Figure 10
Filtering operation of depth image for sky area using semantic segmentation. (Left) Generated depth image (center) Semantic segmentation (right) Filtered depth image



For the semantic segmentation model, xception71_-dpc_cityscapes_trainval[7] function of DeepLab v3+ (Chen et al., 2018) was used. The RGB images of the original GSV were converted to RGBA format with four channels with eight bits per channel and were

resized to 256 × 256 pixels. The value of the R channel of the corresponding estimated depth image was stored to the A channel of the RGBA format image of the GSV, thereby creating an RGBD image. Then, we rotated each RGBD image 36 degrees at the single point, generating nine new images to reduce the bias due to the camera angle. An omnidirectional image was recorded in rectangular form as an equidistant cylindrical projection. Each image was divided into 10 equal parts in the vertical direction, and the left end section of the image was moved to the right end section. We repeated these operations nine times so that the image was taken by rotating the camera 36 degrees. In this way, a total of 1,000 RGBD images were generated.

Learning-to-rank model using DCNN and estimated RGBD images

A learning-to-rank model was used to estimate the evaluation value of streetscape images obtained in the above experiment. Learning to rank is a task of learning the relative order of objects. There are three main methods for learning to rank models based on different combinations of datasets used for learning. We used the principle of RankNet (Burges et al., 2005), a pairwise model for comparing the evaluation values of pairs in a dataset.

We combined the RankNet and DCNN models to input the image data. As the DCNN, we used Resnet-152 (He et al., 2016) pretrained with ImageNet[8] on 1,000 classes, and performed learning by fine-tuning all of its parameters. However, we slightly modified the input and output layers of Resnet-152. The number of channels of the image input layer was changed from three to four, since a RGBD image has four channels. Furthermore, because the output of the RankNet model must be an evaluation value, we changed the final layer of Resnet-152, which is composed of all connected layers through sigmoid functions with 1,000 output nodes, to the layer that returns a scalar value by a simple linear combination.

Because the number of shooting points was only 100, an image shot with a camera angle of 0 degrees was used as the test data for each shooting point. For the remaining 900 images, the images were randomly divided, with images shot at 75 points used as training data and images shot at the remaining 25 points used as validation data. Learning and validation were performed with a configuration of 675 training, 225 validation, and 100 test data points. In the training of this model, for each training image, one additional image was randomly sampled to form a pair and its loss was calculated. Performing this sampling for all training images required one epoch. PyTorch was used as the framework for implementing the learning task. The hyperparameters of RankNet with Resnet-152 are presented below. The number of epochs was 500, the optimization method was Momentum SGD, the moment was 0.9, learning rate was 10^-3, the weight decay was 10^-4, and the mini-batch size was 338.

RESULT OF TRAINING AND VERIFICATION OF METHOD 2

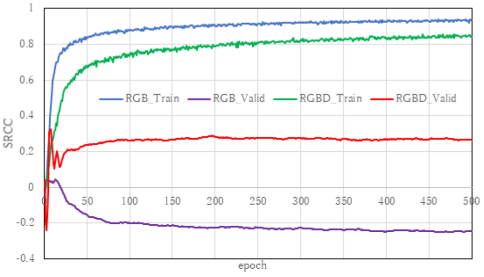


Figure 11
Change in SRCC of training and validation images for each epoch

We trained the RankNet model with the settings described above. The cross entropy of learning pairs is generally used as an error function for RankNet. However, the value differs according to the combination of learning pairs. Therefore, we also evaluated the accuracy of the model using Spearman's rank correlation coefficient (SRCC) for the rank of all images in each dataset. Figure 11 illustrates the change in SRCC in the learning and validation data for the RGB and RGBD images, respectively. The graph demonstrates that the RGB image is more accurate for the training

data. In addition, the validation data indicates that the model trained with RGB images has poor predictive performance and produces the opposite evaluation. In contrast, SRCC of the model trained with RGBD images is 0.3. While this value is not particularly high, the evaluation value of the validation images is somewhat predictable.

The highest SRCC of the validation images trained with RGB and RGBD images was obtained in 13 and 196 epochs, respectively, with the excep-

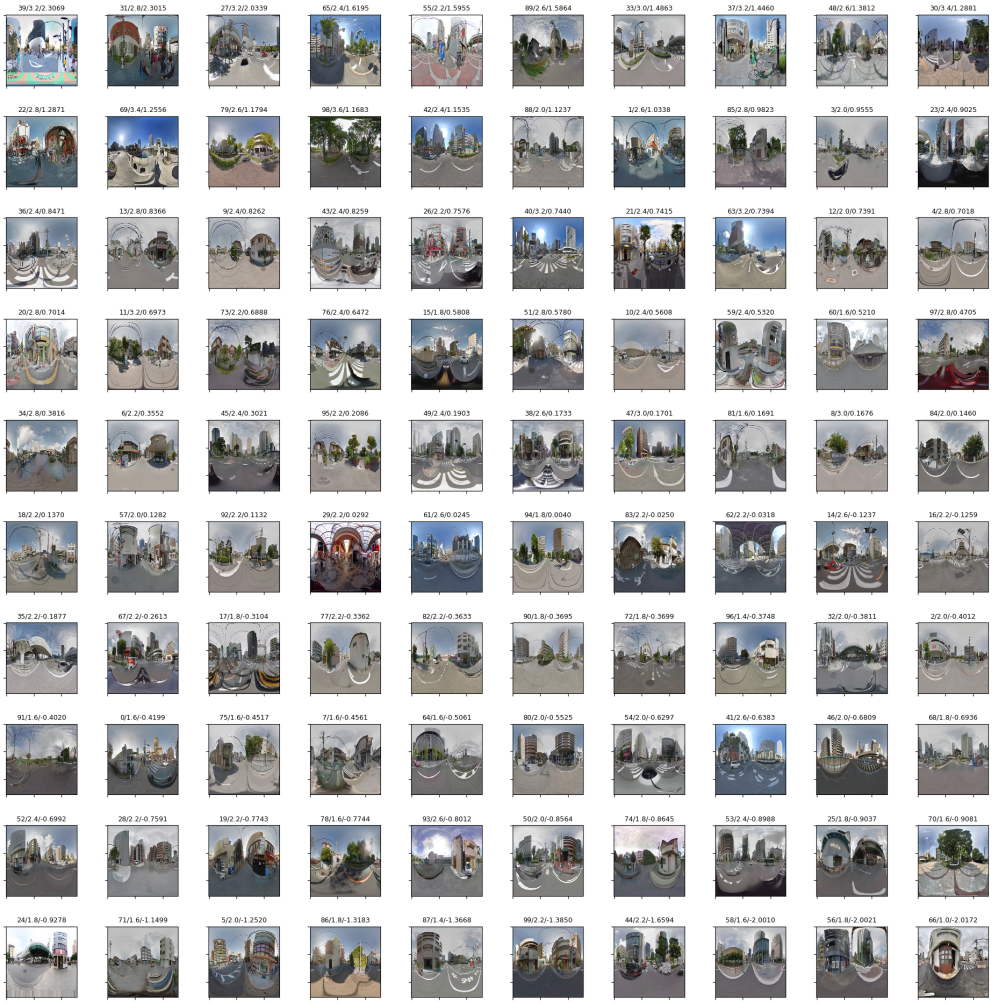


Figure 12
Ranking results of test images by final model trained with RGBD images (ID / actual evaluation value / evaluation value by final model).

tion of the initial unstable training phase. We selected the trained models in these epochs as the best models. Table 2 lists the SRCC of these models with each test image dataset. Because the test images included images shot at the same point as training images, we divided the location used in the training images and the validation images to calculate the SRCC. Whereas the model trained with RGB images exhibited almost no generalization ability, the model trained with RGBD images displayed reasonable performance with a SRCC of 0.41.

Table 2. SRCC and its p-value of each model for test images.

Image type for learning the model	Site for training	Site for validation
RGB	0.743 ($p < 0.001$)	0.045 ($p = 0.830$)
RGBD	0.742 ($p < 0.001$)	0.411 ($p = 0.041$)

Figure 12 presents the test images from the RGBD-trained models sorted in descending order by evaluation value. This figure demonstrates that they are arranged relatively close to the actual evaluation values.

CONCLUSION

In this study, we developed a method for generating omnidirectional depth images from the corresponding omnidirectional RGB images of streetscapes by learning each pair of omnidirectional RGB and depth images created by CG using pix2pix. Models trained with different series of images shot under different site and weather conditions were applied to GSV images to generate depth images. The results suggest that color information alone is insufficient for spatial evaluation, such as that of streetscapes, and that depth information is necessary to improve generalization performance. These result support the findings of a previous study (Takizawa and Furuta, 2017). However, the absolute accuracy of the RankNet model is not very high; it is thus necessary to improve the accuracy of estimated depth images.

ACKNOWLEDGEMENT

This study is partially supported by Grant-in-Aid for Scientific Research (A) and (B).

REFERENCES

Benedikt, M 1979, 'To take hold of space: isovists and isovist fields', *Environment and Planning B*, 6, pp. 47-65

Burges, C, et al. 2005 'Learning to rank using gradient descent', *ICML*, pp. 89-96

Chen, LC, et al. 2018 'Encoder-decoder with atrous separable convolution for semantic image segmentation', *ECCV2018*

He, K, et al. 2016 'Deep residual learning for image recognition', *CVPR2016*, pp. 770-778

Hillier, B and Hanson, J 1989, *The Social Logic of Space*, Cambridge University Press

Isola, P, et al. 2017 'Image-to-image translation with conditional adversarial networks', *CVPR2017*, pp. 5967-5976

Law, S, et al. 2017 'An application of convolutional neural network in street image classification', *GeoAI*, pp. 5-9

Liu, L, et al. 2017, 'A machine learning-based method for the large-scale evaluation of the urban environment', *Computers, Environment and Urban Systems*, 65, pp. 113-125

Ros, G, et al. 2016 'The SYNTHIA dataset: a large collection of synthetic images for semantic segmentation of urban scenes', *CVPR2016*, pp. 3234-3243

Takizawa, A and Furuta, A 2017 '3D spatial analysis method with first-person viewpoint by deep convolutional neural network with omnidirectional RGB and depth images', *Proceedings of eCAADe 2017*, pp. 693-702

[1] https://unity.com/

[2] https://www.nonecg.com/3D-products/tokyo-shibuya/

[3] https://www.nonecg.com/3D-products/japan-8-blocks-34-buildings/

[4] http://www.tanukidigital.com/tenkoku/

[5] https://pytorch.org/

[6] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

[7] http://download.tensorflow.org/models/deeplab_cityscapes_xception71_trainvalfine_2018_09_08.tar.gz

[8] http://www.image-net.org/

Table 2
SRCC and its p-value of each model for test images