YOLO in the Dark - Domain Adaptation Method for Merging Multiple Models -

Yukihiro Sasagawa[1][0000−0002−6313−4941] and
Hajime Nagahara[2][0000−0003−1579−8767]

[1] Socionext Inc., Kyoto, Japan, sasagawa.yukihiro@socionext.com,
[2] Osaka University, Osaka, Japan, nagahara@ids.osaka-u.ac.jp

Abstract. Generating models to handle new visual tasks requires additional datasets, which take considerable effort to create. We propose a method of domain adaptation for merging multiple models with less effort than creating an additional dataset. This method merges pre-trained models in different domains using glue layers and a generative model, which feeds latent features to the glue layers to train them without an additional dataset. We also propose a generative model that is created by distilling knowledge from pre-trained models. This enables the dataset to be reused to create latent features for training the glue layers. We apply this method to object detection in a low-light situation. The YOLO-in-the-Dark model comprises two models, Learning-to-See-in-the-Dark model and YOLO. We present the proposed method and report the result of domain adaptation to detect objects from RAW short-exposure low-light images. The YOLO-in-the-Dark model uses fewer computing resources than the naive approach.

Keywords: Knowledge distillation, Domain adaptation, Object detection

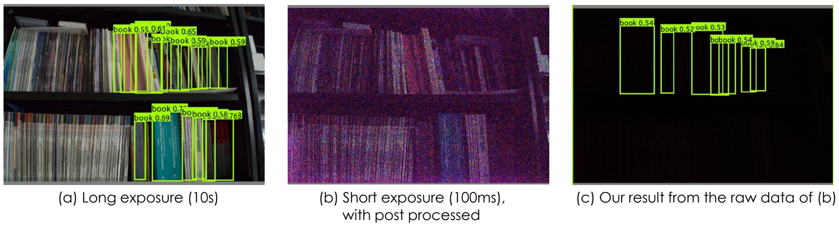
1 Introduction

Performing visual tasks in a low-light situation is a difficult problem. Short-exposure images to not have enough features for visual processing, and the brightness enhancement of the image causes noise that affects visual tasks. In contrast, long-exposure images also contain noise that affects visual tasks owing to motion blur.

In previous work, image processing that handles extreme low-light photography has been developed using an additional dataset (the See-in-the-Dark dataset) [2]. This dataset contains RAW images captured under various exposure conditions. This approach is a straightforward way to create models to perform visual tasks in low-light conditions. However, creating a new dataset requires considerable effort. In particular, end-to-end training for models to perform visual tasks requires many images with annotation.

Knowledge distillation is an excellent way to reuse models trained for other visual tasks [5]. We propose a new method to generate models for performing new visual tasks without the need for an additional dataset. Similar approaches known as unsupervised domain adaptation [1][15] have been proposed. Those methods are effective for changing the domain of a single model (e.g., a classifier). In contrast, our research focuses on merging other models trained on different domains.

We apply the proposed method to achieve object detection in a low-light situation. The well-known object detection model YOLO (You Only Look Once) [10][11] uses public datasets PASCAL VOC and COCO [4][7], and Fig. 1 shows its detection results for low-light images based on the See-in-the-Dark (SID) dataset. Figure 1(a) is the result of a long-exposure (10 s) image, which is sufficient to obtain good results. In contrast, Fig. 1(b) is the result of a short-exposure (100 ms) image that has been brightness enhanced so it the same as that of the long-exposure image. The results of the short-exposure image are degraded.



(a) Long exposure (10s)
(b) Short exposure (100ms), with post processed
(c) Our result from the raw data of (b)

Fig. 1. Object detection in low-light situations. (a) Detection result of a long-exposure (10 s) image. (b) Detection result of a short-exposure (100 ms) image. (c) Our detection result of the RAW data of (b).

We use the Learning-to-See-in-the-Dark (SID) model to handle low-light photography [2]. It improves object detection in the case of short-exposure images. Figure 1(c) shows the result of our method; it can detect objects from the RAW data of the image in Fig. 1(b). Usually, such object detection requires end-to-end training using a dataset containing RAW images with annotations. In contrast, our proposal can import the knowledge of pre-trained models and apply them to models for new visual tasks easily to improve performance. In the remainder of this paper, we present the results of object detection using the proposed method, and we analyze critical elements of the method and discuss directions for future research.

2 Related Work

The lack of datasets in low-light situations has been discussed by researchers. In this section, we review previous research on low-light datasets. Our proposal uses the technique of knowledge distillation, and various methods related to knowledge distillation have been extensively studied in the literature. Hence, we also provide a short review of proposed approaches to knowledge distillation.

Dark image dataset After [2] was presented, [8] discussed datasets for low-light situations. The authors created the Exclusively Dark (ExDark) dataset for research on low-light visual tasks. They found that noise is a notable component in low-light images that affects training. Their results also indicate that denoising improves the edge features of objects, but increases artifacts in the image. The authors also compared features learned by Resnet-50 in bright and low-light images. They initially believed that training data normalization and the progression of data through the layers of a Convolutional Neural Network (CNN) toward high-level abstraction should normalize the data and cause the brightnesses of high- and low-light data to be disregarded because brightness is not a crucial feature for the classification of objects. However, the results of the evaluation indicate that the high- and low-light features are different in the t-SNE embedding space. This result indicates that a model for low-light situations should be trained by an appropriate dataset.

The ExDark low-light dataset is much smaller than the COCO dataset and is too small to create training data for visual tasks in general. However, we refer to this research to compare it with the approach of artificially making each image in the COCO data dataset darker.

Inverse mapping Inverse mapping is a method for image-to-image translation and its aim is to find a mapping between a source ($\mathcal{A}$) and target ($\mathcal{B}$). AEGAN [9] is a well-known approach to obtaining an inverse generator using an AutoEncoder based on generative adversarial nets. Like AutoEncoder, AEGAN contains an inverse generator (IG) and generator (G) and generates image x' from original image x in image domain $\mathcal{A}$ via latent space vector z'. The IG compresses a generated image x into a latent space vector z' and G reconstructs the z' into a new image x'. AEGAN minimizes the difference between generated image x and reconstructed image x'. This structure produces latent space vector z' to represent image-to-image translation.

Invertible AutoEncoder (InvAuto) [14] is another method for image-to-image translation. The translators $F_{\mathcal{AB}}$ ($\mathcal{A}$ to $\mathcal{B}$) and $F_{\mathcal{BA}}$ ($\mathcal{B}$ to $\mathcal{A}$) share InvAuto as part of the encoder (E) and decoder (D). This method is used to convert between the features corresponding to two different image domains ($\mathcal{A}$ and $\mathcal{B}$). Encoder E realizes an inversion of decoder D (and vice versa) and shares parameters with D. This introduces a strong correlation between the two translators.

We use a concept similar to that of AEGAN to create a generative model, as described in Section 3.2.

Hints for knowledge distillation The use of hint information in teacher networks is a well-known approach in knowledge distillation. FitNet [12] is a popular method that uses hint information for model compression. This method chooses hidden layers in the teacher network as hint layers and chooses guided layers in the student network corresponding to the hidden layers. The parameters of the guided layers are optimized by a loss function (e.g., L2 loss) that measures the difference between the outputs of the hint layers and guided layers.

The optimization method for object detection proposed in [3] also uses hint information; this method chooses hint and guided layers and defines the L2 loss as an optimization target. It also refers to the prediction result (the classification and bounding boxes of objects) of a teacher network as a soft target.

We use guided layers during the training, as described in Section 3.3.

3 Proposed model: YOLO in the Dark

3.1 Overview

Figure 2 shows an overview of our method, which merges two models trained in different domains ($\mathcal{A}$ and $\mathcal{B}$). The model for domain $\mathcal{A}$ predicts data Ya from data X. The other model for domain $\mathcal{B}$ predicts data Z from data Yb. Data Ya and Yb are assumed to be the same data type. For example, model $\mathcal{A}$ predicts an RGB image from a RAW image, and the model $\mathcal{B}$ predict an object class and location from the RGB image. After training both models $\mathcal{A}$ and $\mathcal{B}$, this method extracts model fragments with the boundaries of the latent features A and B. The new model is composed of fragments of models $\mathcal{A}$ and $\mathcal{B}$, which are combined through a glue layer.

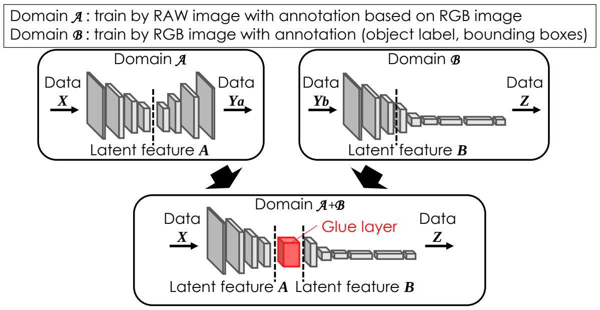

Domain $\mathcal{A}$: train by RAW image with annotation based on RGB image
Domain $\mathcal{B}$: train by RGB image with annotation (object label, bounding boxes)
Domain $\mathcal{A}$
Data
X
Data
Ya
Latent feature A
Domain $\mathcal{B}$
Data
Yb
Data
Z
Latent feature B
Domain $\mathcal{A}$+$\mathcal{B}$
Data
X
Glue layer
Data
Z
Latent feature A
Latent feature B

Fig. 2. Our method of domain adaptation, which merges two models trained in different domains $\mathcal{A}$ and $\mathcal{B}$.

This glue layer helps transform latent feature A to latent feature B in each model fragment. The SID model performs well on low-light images [2], so we use the SID model for model $\mathcal{A}$. We also use the object detection model YOLO [10][11] for model $\mathcal{B}$.

3.2 Generative model for domain adaptation

Training the glue layer requires additional data for domain $\mathcal{A} + \mathcal{B}$; however, creating a dataset requires considerable effort. Our method defines a generative model for training the glue layer using knowledge distillation.

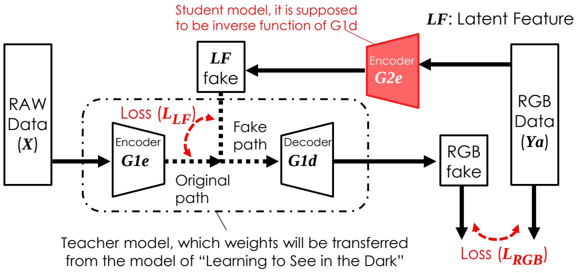

Student model, it is supposed to be inverse function of G1d
LF: Latent Feature
LF fake
Encoder G2e
RAW Data (X)
RGB Data (Ya)
Loss (L_{LF})
Fake path
Encoder G1e
Decoder G1d
Original path
RGB fake
Teacher model, which weights will be transferred from the model of "Learning to See in the Dark"
Loss (L_{RGB})

Fig. 3. Scheme for knowledge distillation. The red part $G2e$ is the student model.

Figure 3 illustrates the scheme of knowledge distillation for the generative model. The generative model outputs the latent features of A from data Ya, as described in Fig. 2. The SID model is an encoder–decoder structure, so the generative model is the inverse function of the decoder. In Fig. 3, encoder–decoder $G1e$-$G1d$ is a teacher model and train student model $G2e$ by feeding RAW and RGB image pair from the SID dataset and SID model $G1e$-$G1d$.

This training uses the loss between RGB data and "fake" RGB data reconstructed via the $G2e$-$G1d$ combination, expressed as follows:

$$L_{RGB} = \|RGB_{data} - RGB_{fake}\|_1 \tag{1}$$

The training also uses another loss between the latent features LF from $G2e$ and latent features from $G1e$, as follows:

$$L_{LF} = \sum_i \|{LF^i}_{G2e} - {LF^i}_{G1e}\|_1, \tag{2}$$

where i is the index number of the layers starting from the outputs of each encoder ($G2e$ or $G1e$).

These two loss functions help define $G2e$ as the inverse function of $G1d$, and the total loss function is as follows:

$$L_{total} = L_{RGB} + L_{LF}. \tag{3}$$

Figure 4 shows the structure of the glue layer for the latent features of the SID encoder. Figure 4 (a) is the SID network structure. As explained in [2], the SID network is based on U-net [13], which is composed of an encoder and a decoder. The encoder extract features using convolution and pooling layers. The pooling layer behaves as a low-pass filter for spatial frequency so that the features contain different frequency information as a result of each pooling layer. The SID Encoder has four levels of features corresponding to the pooling scales 1/1, 1/2, 1/4, and 1/8. Each layer in the encoder feeds the latent features to the corresponding layers in the decoder. The glue layer must be sufficiently expressive with respect to frequency information for the subsequent network (i.e., the object detection network).

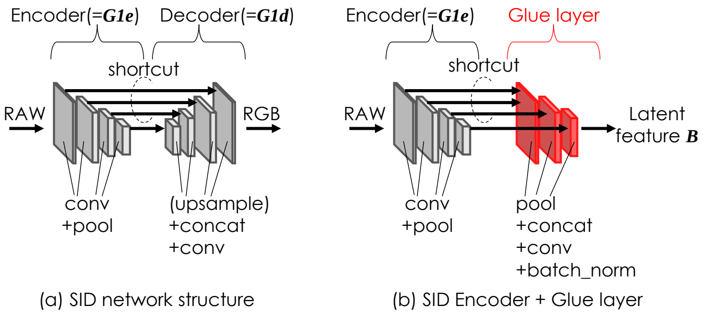


Fig. 4. Structure of the glue layer. (a) SID network structure. (b) Glue layer structure after the SID encoder.

Figure 5 shows the reconstructed RGB images using the latent features of the SID encoder. Figure 5 (a) illustrates images reconstructed using all features. The images have a Peak Signal-to-Noise Ratio (PSNR) of 31.81 and Structural Similarity (SSIM) of 0.752 with respect to the original images. Figures 5 (b), (c), and (d) are images reconstructed using fewer features, which removes high spatial frequency information. The quality of these images is worse than that of Fig. 5 (a). To detect objects, the detailed shape of the object must be identified, so we decided to use all latent features for the glue layer.

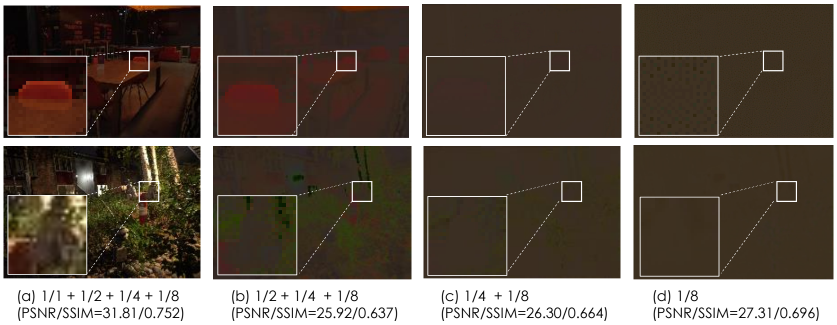


Fig. 5. Reconstruction using latent features, with the quality metrics (PSNR/SSIM) to original images. Reconstruct results using (a) 1/8-, 1/4-, 1/2-, and 1/1-scale features, (b) 1/8-, 1/4-, and 1/2-scale features, (c) 1/8- and 1/4-scale features, and (d) 1/8-scale features.

Figure 4 (b) shows the glue layer structure following the SID encoder. The glue layer is composed of pooling, concatenation, convolution, and batch normalization. The pooling and concatenate functions help gather latent features. The convolution and batch normalization functions help convert a new latent feature for domain $\mathcal{B}$.

The knowledge in the generative model is distilled, as described in Fig. 3, according to the structure of the glue layer. Figure 6 shows the RGB images reconstructed using the distilled knowledge. The RGB images from the $G2e$-$G1d$ combination in Fig. 6(b) and the RGB images generated by the SID model in Fig. 6(a) seem similar. The SID-model generated RGB images have a PSNR of 30.18 and SSIM of 0.917 with respect to the "fake" RGB images. These values indicate good image similarity, so we conclude that the behavior of $G2e$ is sufficiently similar to that of the inverse function of $G1d$.

We also fine-tune the $G2e$ to improve the transformation of latent feature A to latent feature B in each model fragment. We focused on the result of the classifier network in the YOLO model to optimize the generative model. According to [10], the YOLO model contains a classifier network at the beginning of the network itself. This classifier network learns the feature map for processing in the succeeding detection network so it should effectively optimize the generative model corresponding to the SID dataset. We use cosine similarity as the finetuning loss L_{G2e-FT} between the results of the classifier network via $G2e$ ($LF_{G2e-cls}$) and the original YOLO ($LF_{YOLO-cls}$) as follows:

$$L_{G2e-FT} = cos(\overrightarrow{LF_{G2e-cls}}, \overrightarrow{LF_{YOLO-cls}}), \tag{4}$$

where $\overrightarrow{LF_{G2e-cls}}$ and $\overrightarrow{LF_{YOLO-cls}}$ are vectors reshaped from the feature tensors of the classifier networks. In addition, $LF_{G2e-cls}$ and $LF_{YOLO-cls}$ are described in Fig. 7.

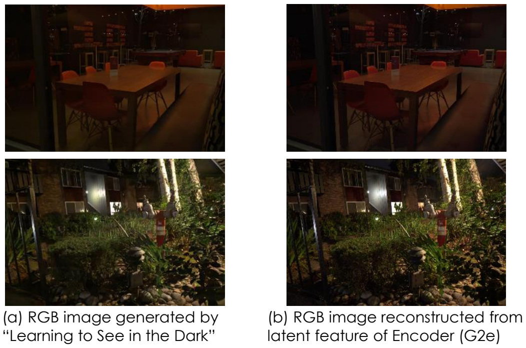
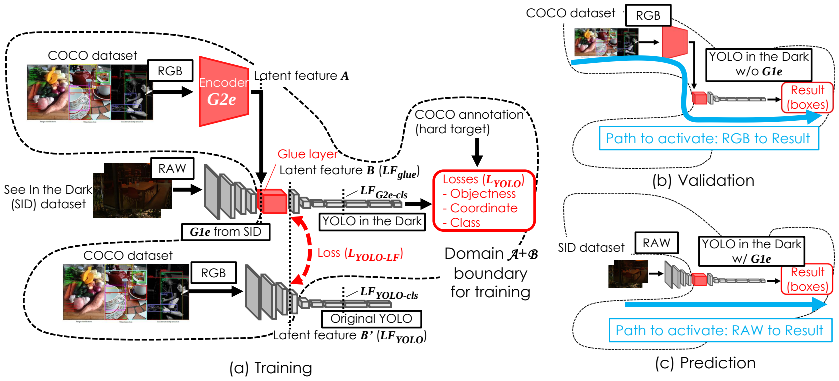

(a) RGB image generated by
"Learning to See in the Dark"
(b) RGB image reconstructed from
latent feature of Encoder (G2e)

Fig. 6. Reconstructed RGB images. (a) Image generated by the SID model, which is equivalent to $G1e$-$G1d$. (b) Image generated by $G2e$-$G1d$.

3.3 Training environment

Figure 7 shows the training environment for the proposed YOLO-in-the-Dark model $\mathcal{A} + \mathcal{B}$. Figure 7(a) shows a complete view of the environment, where the dotted boundary shows the parts used for training the new model. The glue layer is the target of training, which uses the RGB data via the encoder $G2e$ generated by knowledge distillation. The training environment uses the original YOLO model, which uses the same RGB data as $G2e$. We use the COCO dataset [7] for training.



(a) Training

(b) Validation

(c) Prediction

Fig. 7. Training environment of the YOLO-in-the-Dark model. (a) Complete view of the environment. (b) Validation behavior using the RGB data for inference. (c) Prediction behavior using the RAW data for inference.

During the training phase, the glue layer is optimized according to the loss functions. The first loss function uses the annotation (object coordinates and class) of the dataset, which is the same as in the original YOLO scheme [11].

The other loss function is based on a latent feature in the original YOLO model, which is the L2 loss between latent feature B in the YOLO-in-the-Dark model and the latent feature B' in the original YOLO, expressed as follows:

$$L_{YOLO-LF} = \|LF_{glue} - LF_{YOLO}\|_2. \tag{5}$$

Loss $L_{YOLO-LF}$ works as a regularization term so that the total loss uses the second loss with a coefficient as follows:

$$L_{total} = L_{YOLO} + \lambda L_{YOLO-LF}, \tag{6}$$

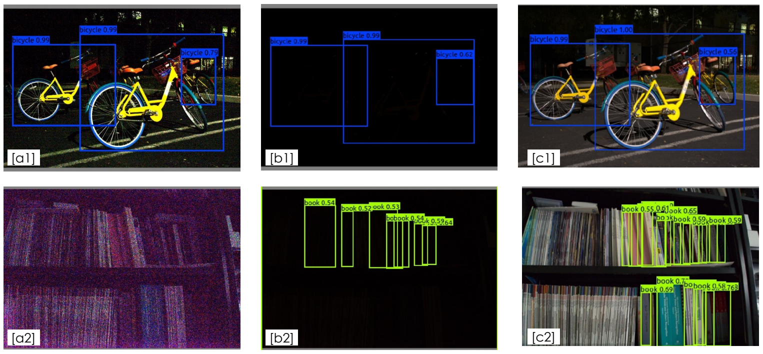
where L_{YOLO} is the same as the loss function used in the original YOLO scheme [11].

Figure 7(b) shows the dataflow during validation. The validation uses the same path as training, which uses the RGB data and evaluates enough samples from the dataset to confirm the glue layer is behaving correctly. Figure 7(c) shows the dataflow during prediction. The prediction uses the other path, using the RAW data via the encoder $G1e$ transferred from the SID model. This stage is for evaluating the proposed YOLO-in-the-Dark model, which will improve object detection in short-exposure RAW images.

4 Experiments

4.1 Object detection in RAW images

Figure 8 shows the object detection results for the SID dataset. Figure 8 (a) is the result obtained by the original YOLO model, which used a brightness enhanced RGB image. The brightness enhancement of the RGB image makes it easier for the original YOLO model to detect objects. The original YOLO model detects the objects in image a1 well. However, this model cannot detect the objects in image a2. This is because the brightness enhancement adds noise and affects the inference. Our proposed YOLO-in-the-Dark model detects the objects in the RAW images directly. The detection results are shown in images b1 and b2. Images c1 and c2 are the baseline detection results in which the original YOLO model uses the SID ground truth (long-exposure) image. In image b1, the proposed model performs as well as the original YOLO model (image a1). In addition, the proposed model can detect objects in image b2.



(a) Original YOLO model, refer to post-processed RGB image.

(b) Our work, refer to RAW image. (boxes on input image)

(c) Original YOLO Baseline, refer to SID ground truth (long-exposure) image.

Fig. 8. Detection results. (a) Results of the original YOLO model on a brightness enhanced RGB image. (b) Results of our proposed model on a RAW image. (c) Baseline detection results of the original YOLO model on a long-exposure image.

The SID dataset [2]contains both indoor and outdoor images, and the illuminance at the camera in the indoor scenes is between 0.03 lux and 0.3 lux. Hence, the results for image b2 indicate that the YOLO-in-the-Dark model can handle scenes illuminated by less than 1 lux.

As explained in Section 1, the YOLO-in-the-Dark model uses the encoder in SID at the front end to process RAW images. The new model creates a latent feature using the SID encoder, which processes the low-light image and outputs the results to the back end of the YOLO model via the glue layer. This improves object detection performance in low-light situations.

We also evaluated other images in the SID dataset. Figure 9 shows the prediction results (F-measure) using the SID dataset categorized by the size of the bounding boxes. We created annotation data for this evaluation by detecting objects from reference images using the original YOLO model. The reference images were captured under a long exposure so that the original YOLO could detect the objects easily. We used these detection results as the ground truth for this evaluation.

Figure 9 shows the results for the original YOLO model, which detected objects in brightness-enhanced RGB images. Figure 9 also shows the YOLO model trained by the dark COCO dataset, in which the brightness was scaled to emulate low-light situations. This trained YOLO performs worse than the original YOLO model. Previous work [8] has found that noise is a notable component in low-light images, and it hence affects training. The study [8] also found that denoising improves the edge features of the objects but increases the artifacts. SID [2] should hence be a good solution to this problem. In addition, Figure 9 shows the results for the YOLO-in-the-Dark model. The F-measure is better for all sizes of bounding box than those of the first two models. The mAP (at an Intersection over Union (IOU) threshold of 50%) improved by 2.1 times (0.26 → 0.55) compared with the original YOLO model. The latent features defined in Section 3.2 seem to be sufficient to detect any size of object, and the distillation of knowledge by the generative model has been successful.

Finally, Figure 9 shows the results for the SID+YOLO model, which is a simple combination of SID and YOLO (the naive approach). Like the YOLO-in-the-Dark model, the naive approach uses RAW images, but it generates an RGB image using SID. The YOLO-in-the-Dark model reuses parameters of both SID and YOLO so that it ideally should achieve the same performance as the SID+YOLO model. Figure 9 instead indicates that the YOLO-in-the-Dark model has a performance that is close but not equal to that of the SID+YOLO model. We finetuned both the glue layer and the generative model to improve the transformation of latent feature A to latent feature B in each model fragment. Figure 9 also shows the results for the YOLO-in-the-Dark model without the finetuning. We discuss this point in Section 4.2.

We also evaluated the detection performance in low-light situations. The SID dataset does not cover various levels of illuminance, so we collected additional indoor images that contain objects (e.g., cars, fruit, and animals). Each image was captured using the same camera settings (f/5.6 and ISO-6400). Both the original YOLO model and the proposed YOLO-in-the-Dark model can detect objects at an illuminance of 0.055 lux with similar exposure times (case (a)). In contrast, the original YOLO model requires a longer exposure time (810 ms) when the illuminance is 0.013 lux (case (b)). The YOLO-in-the-Dark model still detects objects even when the exposure time is shorter (810 → 333 ms). The minimum exposure time indicates the sensitivity in low-light situations. According to the result of case (b), the YOLO-in-the-Dark model can reduce exposure time by a factor of 0.4; this means that the sensitivity is improved by

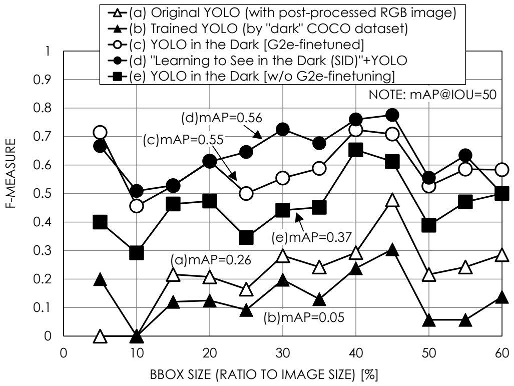


Fig. 9. Prediction results (F-measure) using the SID dataset for the original YOLO model, the YOLO model trained by dark COCO, the YOLO-in-the-Dark model, the SID+YOLO model, and the YOLO-in-the-Dark model without finetuning the generative model.

2.4 times when compared with the sensitivity of the original YOLO model using brightness-enhanced RGB images.

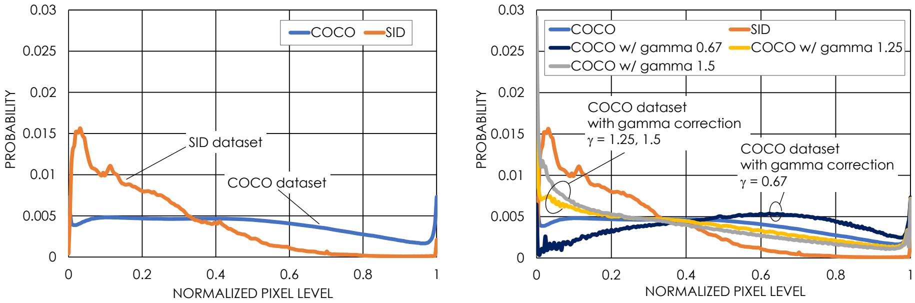
4.2 Ablation study

As explained in Section 3.2 and 3.3, the training environment of the YOLO-in-the-Dark model uses the COCO dataset with the generative model trained by the SID dataset. Our method supposes that the encoder $G2e$ outputs the latent feature A from the COCO dataset, which emulates the relationship between a pair of RAW and RGB images in the SID dataset. In this section, we describe a sequence of controlled experiments that evaluate the effect of different elements in the training environment.

Input images Figure 10(a) shows the histograms of the average image in both the COCO and SID datasets. Images in the SID dataset tend to have low pixel levels; this is because of the low-light situations. The distribution of pixel levels for the images in the COCO dataset differs from that of the SID dataset, and this difference might affect encoder $G2e$.

We evaluate the training with different preprocessing parameters (gamma correction) for the input images. Figure 10(b) shows the histograms of the average image in the COCO dataset with gamma correction. A large gamma leads to low pixel levels, as in the SID dataset. Figure 11 reports the results for different levels of gamma correction of 0.67, 1.25, and 1.5.

The histograms of the preprocessed COCO dataset show that gamma correction with a value of 1.25 or 1.5 causes the histogram to become close to the histogram of the SID dataset. Hence, gamma correction of the training images should improve the results of the model. When gamma is 0.67, Fig. 11 shows



(a) Histograms for each dataset

(b) Histograms for each gamma correction

Fig. 10. Histograms of average dataset images. (a) Histograms for the average image of each dataset and (b) histograms after each image has been gamma corrected.

that the mAP slightly degrades (0.37 → 0.36). When gamma is 1.25, the mAP improves (0.37 → 0.40), and when 1.5, the mAP does not change.

These results show that gamma correction can mitigate the effects due to differences in the datasets. However, the model is sensitive to the value of gamma, and this leads to side effects. Our evaluation indicates that the gamma correction requires fine adjustment corresponding to the histogram of pixel levels in the dataset.

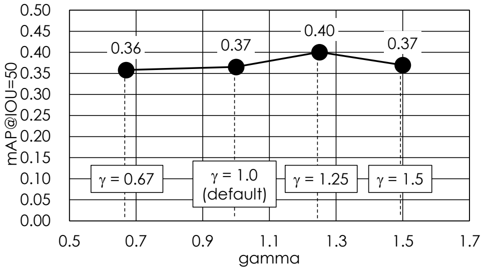


Fig. 11. Detection results (mAP) corresponding to the level of gamma correction in the training images.

Data augmentation Data augmentation is an effective way to improve the generalization of the model [6]. We also evaluate the effect of the data augmentation as well as gamma correction for input images.

Table 1 reports the case in which the data has been augmented by changing brightness and contrast. This approach randomly changes the brightness and contrast by 10% of the maximum pixel levels in each image. The mAP slightly degrades (0.37 → 0.35) compared with the default condition (no augmentation). According to previous reports, that found that the results are sensitive to the value of gamma, data augmentation may have a negative effect on this training. The proposed method reuses the pre-trained model, which had been generalized well. Hence, it needs generalization only for the glue layer. As a result, data augmentation does not seem effective for this method.

Table 1. Results for data augmentation conditions (mAP).

Condition	mAP@IOU=50
Default dataset processing	
(No augumentation)	0.37
Contrast/brightness augmentation	0.35

Finetuning the generative model As described in Section 3.2, we finetune the generative model to improve the training of the glue layer. We use the finetuning loss function L_{G2e-FT} between the results of the classifier network via $G2e$ ($LF_{G2e-cls}$) and the original YOLO ($LF_{YOLO-cls}$), as described in Fig. 7. This finetuning optimizes $G2e$ with respect to both features $LF_{G2e-cls}$ and $LF_{YOLO-cls}$ so that the optimized $G2e$ can output a better latent feature A. This helps optimize the glue layer. Table 2 reports the L_{G2e-FT} and mAP, corresponding to the cases (c) and (e) in Figure 9. The result shows that finetuning the generative model substantially improves the mAP with a lower cosine similarity.

Table 2. Results for $G2e$ finetuning L_{G2e-FT} and mAP under each condition.

Condition	L_{G2e-FT}	mAP@IOU=50
Default dataset processing		
(not finetuned)	0.21	0.37
$G2e$ finetuned	0.15	0.55

Computing resources The other contribution of this method is that it reduces the amount of computing resources required. Table 3 compares the resources used by the proposed method and the SID+YOLO model (MAC operations) to process an 832 × 832 RAW image created by resampling from the original RAW image (4,240 × 2,832). Our method can omit the SID encoder by merging models via latent features. The SID decoder is computationally costly (192.63 GMACs), so the glue layer reduces the total amount of computational resources needed.

Table 3. Comparison of the computing resources used by the proposed and naive approaches (on a 832 × 832 RAW image).

Processing category	MAC operations [GMACs]	
	SID+YOLO	Our work
SID Encoder	45.45	45.45
SID Decoder	192.63	N/A
Glue Layer	N/A	36.68
YOLO	32.71	31.77
Total	270.79	113.90

5 Conclusion

We proposed a method of domain adaptation for merging multiple models that is less effort than creating an additional dataset. This method merges models pre-trained in different domains using glue layers and a generative model, which outputs latent features to train the glue layers without the need for an additional dataset. We also propose a generative model that is created by knowledge distillation from the pre-trained models. It also enables datasets to be reused to create latent features for training the glue layers.

The proposed YOLO-in-the-Dark model, which is a combination of the YOLO model and SID model, is able to detect objects in low-light situations. Our evaluation result indicates that the YOLO-in-the-Dark model can work in scenes illuminated by less than 1 lux. The proposed model is also 2.4 times more sensitive than the original YOLO model at 0.013 lux. Simply combining SID and YOLO (the naive approach) also improves object detection in low-light situations, and this naive approach demonstrates the ideal performance of the YOLO-in-the-Dark model. The performance of our model still needs to be improved so it is closer to this ideal. We also presented an evaluation of the YOLO-in-the-Dark model under various conditions. Preprocessing training input images and fine-tuning the generative model are effective in improving the optimization of the glue layer.

The other contribution of this method is the reduction in computing resources. In contrast to the naive approach, our method can omit the SID decoder by merging models via latent features. The SID decoder is computationally expensive (192.63 GMACs), so the glue layer can reduce the total amount of computational resources required.

In future work, we plan to apply this method to other tasks including multimodal tasks. We hope that this method can be applied to various models using public datasets. It will extend the functionalities of models more efficiently.

References

1. Chang, W.G., You, T., Seo, S., Kwak, S., Han, B.: Domain-specific batch normalization for unsupervised domain adaptation. In: CVPR2019 (2019)
2. Chen, C., Chen, Q., Xu, J., Koltun, V.: Learning to see in the dark. In: CVPR2018 (2018)
3. Chen, G., Choi, W., Yu, X., Han, T., Chandraker, M.: Learning efficient object detection models with knowledge distillation. In: Advances in Neural Information Processing Systems 30. pp. 742–751 (2017)
4. Everingham, M., Gool, L.V., Williams, C., Winn, J., Zisserman, A.: The pascal visual object classes challenge 2012 (voc2012) results. In: VOC2012 (2012)
5. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network (2015), arXiv:1503.02531
6. Krizhevsky, A., Sutskever, I., Hinton, G.: Imagenet classification with deep convolutional neural networks. In: Proceedings of the Advances Neural Information Processing Systems 25 (NIPS). pp. 1097–1105 (2012)
7. Lin, T.Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C.L., Dollar, P.: Microsoft coco: Common objects in context (2014), arXiv:1405.0312
8. Loh, Y.P., Chan, C.S.: Getting to know low-light images with the exclusively dark dataset. Computer Vision and Image Understanding **178**, 30–42 (2019). https://doi.org/https://doi.org/10.1016/j.cviu.2018.10.010
9. Luo, J., Xu, Y., Tang, C., Lv, J.: Learning inverse mapping by autoencoder based generative adversarial nets (2017), arXiv:1703.10094
10. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR2016 (2016)
11. Redmon, J., Farhadi, A.: Yolov3: An incremental improvement (2018), arXiv:1804.02767
12. Romero, A., Ballas, N., Kahou, S.E., Chassang, A., Gatta, C., Bengio, Y.: Fitnets: Hints for thin deep nets (2014), arXiv:1412.6550
13. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: MICCAI (2015)
14. Teng, Y., Choromanska, A., Bojarski, M.: Invertible autoencoder for domain adaptation (2018), arXiv:1802.06869
15. Xie, S., Zheng, Z., Chen, L., Chen, C.: Learning semantic representations for unsupervised domain adaptation. In: ICML2018 (2018)