Exploring the power of lightweight YOLOv4

Chien-Yao Wang[1], Hong-Yuan Mark Liao[1,2], I-Hau Yeh[3], Yung-Yu Chuang[4], and Youn-Long Lin[5]
[1]Institute of Information Science, Academia Sinica, Taiwan
[2]Department of Computer Science and Information Engineering, Providence University, Taiwan
[3]Elan Microelectronics Corporation, Taiwan
[4]Department of Computer Science and Information Engineering, National Taiwan University, Taiwan
[5]Department of Computer Science, National Tsing Hua University, Taiwan
kinyiu@iis.sinica.edu.tw, liao@iis.sinica.edu.tw, ihyeh@emc.com.tw,
cyy@csie.ntu.edu.tw, and ylin@cs.nthu.edu.tw

Abstract

Research on deep learning has always had two main streams: (1) design a powerful network architecture and train it with existing learning methods to achieve the best results, and (2) design better learning methods so that the existing network architecture can achieve the best capbility after training. In recent years, because mobile device has become popular, the requirement of low power consumption becomes a must. Under the requirement of low power consumption, we hope to design low-cost lightweight networks that can be effectively deployed at the edge, while it must have enough resources to be used and the inference speed must be fast enough. In this work, we set a very ambitious goal of exploring the power of lightweight neural networks. We utilize the analysis of data space, model's representational capacity, and knowledge projection space to construct an automated machine learning pipeline. Through this mechanism, we systematically derive the most suitable knowledge projection space between the data and the model. Our method can indeed automatically find learning strategies suitable for the target model and target application through exploration. Experiment results show that the proposed method can significantly enhance the accuracy of lightweight neural networks for object detection. We directly apply the lightweight model trained by our proposed method to a Jetson Xavier NX embedded module and a Kneron KL720 edge AI SoC as system solutions.

1. Introduction

Low power computer vision is one of the most concerned topics of academia and industry today, mainly because nowadays there are many applications deployed on artificial intelligence of things (AIoT). On edge devices, audio applications usually take 16 kHz, 16-bit depth audio as input. As for computer vision applications, the common input is a video with 480p (SD) to 1080p (full HD) resolution. If the frame rate is 15∼30 FPS and calculated with 8-bit color depth, it is about 150 to 2000 times more than the amount of audio data. Therefore, compared with the more mature low power audio technology, it is relatively more challenging to develop a fast and effective low power computer vision system.

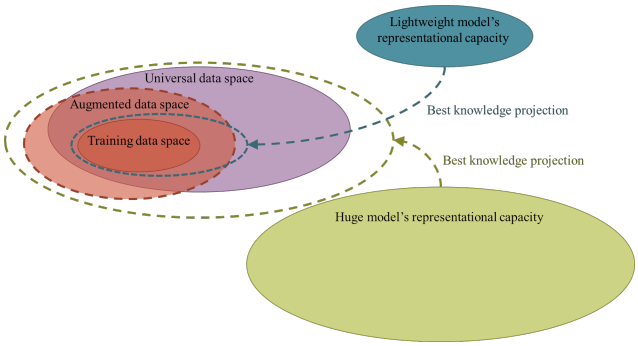


Figure 1. Unlike the huge model, which is an over-parameterized model, a lightweight model is an under-parameterized model, its representational capacity can only cover a sub-space of the universal data space.

Due to the need of operating on equipment with extremely limited resources, the lightweight neural network becomes the best option to achieve the required speed and accuracy. In order to obtain an appropriate lightweight neural network for target device, network architecture search (NAS) [12, 9, 20, 19] is the most direct and effective method. However, the network searched by NAS is usually not universally applicable to the tasks that are not involved in training the NAS, and not all target devices have interfaces suitable for training NAS. Model pruning [10, 23] is another technique that is often used to obtain lightweight neural networks, and this technique can be applied to a

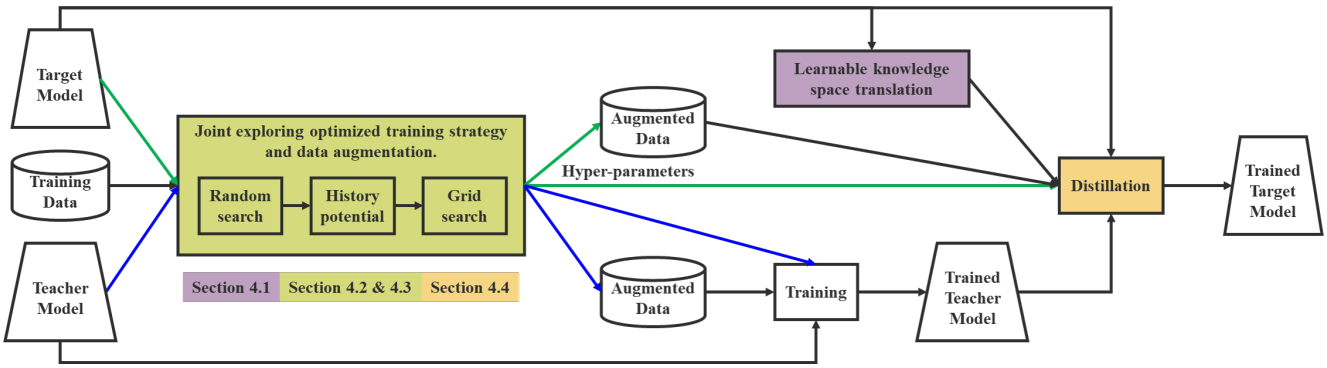


Figure 2. Block diagram of the proposed method. It mainly includes three procedures: (1) Learnable knowledge space translation (4.1); (2) Joint exploring optimized data augmentation method (4.2) and training strategy (4.3); and (3) Exploring suitable knowledge distillation space (4.4).

trained model. However, model pruning usually produces unstructured architecture, which makes the inference speed of the target device unable to reach the expected level. As for the structured model pruning [38, 18, 22] approaches, although they can steadily increase the inference speed, due to additional restrictions, the degree of pruning is not significant. Meanwhile, structured model pruning does not guarantee to maintain the best utilization of target device resources. So, instead of removing weights that do not function, why not activate them? Luo *et al*. put forward this view in [30]. In this work, our main purpose is to explore the power of lightweight neural networks. We propose an automated machine learning method by analyzing the relationship between the representational capacity of lightweight models and the data space of the target task. This will enable a lightweight model to maximize its effectiveness and find the best knowledge projection results, as shown in Figure 1.

Goodfellow *et al*. [8] once said that due to the imperfection of the optimization method, the effective capacity of a model after training is usually less than the representational capacity of the model itself. Therefore, it is very difficult to find the best knowledge projection space of a model. In view of it, we combine multiple aspects to design an automated machine learning pipeline for exploring the power of lightweight neural networks, as shown in Figure 2. This figure contains the three main procedures proposed in this work, including (1) joint exploring optimized training strategy and data augmentation (green), (2) distillation (yellow), and (3) learnable knowledge space translation (purple).

Based on the analysis of the data space and representational capacity of a model, we hypothesize that the expected value of the distribution between the above two should be jointly calculated to represent the theoretical effective capacity. Therefore, for the first procedure, we will perform joint hyper-parameter search on data augmentation and training strategy.

As for the second procedure, we try to bridge the best knowledge projection space of the target model and the teacher model. We use the method proposed in the first procedure to search for the training strategy on the target model, and find a concise over-complete sub-space that is very close to the knowledge projection space of the target model, which is located in the knowledge projection space of the teacher model. We use the knowledge projection of the teacher model in this sub-space to distill the knowledge of the target model. In this way, we can optimize the process by considering the generalization and the fidelity issues. The above issues are the common problems in the existing knowledge distillation method raised by Stanton *et al*. [33] in 2021.

While most general training methods can only find a point-based solution, the learning method proposed by Wortsman *et al*. [39] tried to find a line-segment-based solution. In this work, we hope to find a distribution-based solution using the third procedure. By introducing multiple sets of learnable implicit representations in the random exploration in the knowledge projection space, we can determine multiple approximate solutions, which form a convex region. Using our proposed method to train the model and perform knowledge distillation can make the system achieving stronger degree of generalization. Our contribution includes:

1. We propose an automated machine learning pipeline to study the power of lightweight neural networks.
2. We propose a learnable knowledge space translation module for finding subspace of multiple approximate solutions in the solution space.
3. We discuss how to effectively find a suitable knowledge distillation space, and prove that the use of the teacher model in the subspace around the target model's knowledge projection space can achieve better knowledge distillation.
4. The proposed method empowers a SOTA lightweight object detector to significantly improve average precision (AP) on MSCOCO dataset by 1.2%.

2. Related work

In this section, we will survey related literature, including representational capacity and trainability of DNN (2.1). We will also discuss the model distillation of DNN during various stages of training (2.2). Finally, we survey data augmentation methods that can augment the training data space (2.3).

2.1. Training-free network architecture search

Training-free network architecture search (NAS) relies on, the representational capacity of the model [26, 17, 41] and the trainability of the computing unit [15, 40].

In [26], Montufar *et al*. calculated the maximum number of deep neural network (DNN) that can cut the hyperplane into linear regions. Under ideal conditions, one can assign each of the above linear region to a category. Therefore, we can use the number of linear regions that a DNN can separate into to estimate the representational capacity of a model. [41] further extend the above mentioned hyperplane separation of DNN to the case of convolutional neural network (CNN). The neural tangent kernel proposed in [15] proves that an infinitely wide neural network can be regarded as a regression model that undergoes specific feature conversion, and many finitely wide neural networks have similar properties. Xiao *et al*. [40] estimated the trainability of neural networks with the mean-field theory based on neural tangent kernel. Recently Mellor *et al*. [24] designed a training-free NAS method to calculate the representational capacity of the neural network in a pre-defined search space. In [3], Chen *et al*. took into account the two characteritics of reprensentational capacity and trainability at the same time.

2.2. Model distillation

In the model distillation part, we will discuss knowledge distillation and model pruning. The concept of knowledge distillation is proposed by Hinton *et al*. [11], and the main idea is to let the student model learn a soft label of the teacher model. It [11, 2, 37, 31, 32, 42, 28] is a training method that condenses the knowledge learned by a large teacher model into the essence and passing it to a small student model. Therefore, it can be considered as indirect model distillation since it does not directly apply compression on the model. Model pruning [10, 38, 18, 23, 22] removes useless weights or connections on the trained model. It can be regarded as direct model distillation, because it executes compression directly on model.

Chen *et al*. [2] proposed to simultaneously distill knowledge of feature map and output for the object detection task. Shen *et al*. [31, 32] conducted ensemble knowledge distillation for multiple teacher models. Recently, knowledge distillation has also been widely used in the self-supervised learning field [42, 28].

Model pruning can be divided into two categories, i.e., unstructured pruning and structured pruning. The former allows unnecessary weights to be removed at will, while the latter adds restrictions so that the pruned model can still use conventional computing units. Unstructured pruning methods [10, 23] usually have a higher model compression rate, but the resulting DNN architecture is only suitable for sequential operation equipment. The structured pruning method [38, 18, 22] generally cuts the entire channel or layer, so it can effectively improve the inference speed of a model. However, its compression rate is not as good as the unstructured one.

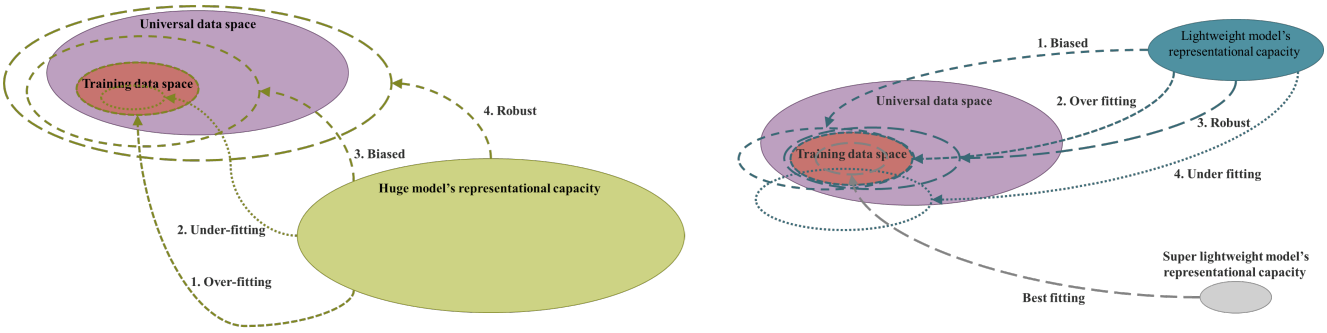
2.3. Data augmentation

There are two types of data augmentation: general data augmentation [44, 43, 7] and generative models for data augmentation [13, 34, 6, 29]. The former is applied directly on the data, using artificial criteria or metadata processing, while the latter uses generative models such as autoencoders and generative adversarial networks (GANs) to augment the dataset.

General data augmentation is roughly divided into two types. The first one is to make pixel-level changes to the image content, such as adjusting color, constract, jittering, rotation, and flipping, and so on. The second is to merge information of multiple images to make patch-level changes. For example, the work proposed in [44] superimposes two images to generate soft labels, and [43] proposed to cover the patches obtained by cropping different images on source image for data augmentation. Furthermore, the work in [7] uses instance segmentation labels to enhance the robustness of data augmentation.

A generative model-based data augmentation method generates new images for training. Among them, the autoencoder-based method [34] generates new data using variability to the existing data, while the GAN-based method [13] samples from a latent space to generate new data. In [29], Sandfort *et al*. used CycleGAN [45] for data augmentation, while Geirhos *et al*. [6] used style transfer technology [14] to generate augmented images of various artistic styles to reduce the shape bias.

In addition to the more commonly used methods, there are some methods that create search space from the existing data augmentation methods to automatically search for the policy of data augmentation [4, 46].



(a) Knowledge projection of huge model **(b) Knowledge projection of lightweight model**

Figure 3. Knowledge projection space of huge model and lightweight model. For huge model, its representational capacity can cover the universal data space. However, for lightweight model, the best knowledge projection is still under-fitting the universal data space.

3. Learning space and model capacity analysis

Figure 3 shows the knowledge space and data space obtained by models of different scales after learning. We will analyze the learning space and representational capacity of the model. In the data space analysis (3.1), we will analyze the theoretical numerical space, the statistics-based space calculated based on natural images, and the real data space in practical applications. As for the space of model's representational capacity (3.2), we will calculate the space of model's representational capacity by combining the model's representational capacity, the trainability of the computing unit, and the analysis of the data space. Finally, in the analysis of the knowledge projection space (3.3), we will explore the relevance between models, and between the model and data in the learning process.

3.1. Data space analysis

Theoretical Data Space: Given an image with width W and height H, if each pixel in the image has C values, for example, each pixel in an RGB image is represented by three values. These values are all quantized into Q levels of values. For example, a common 8-bit depth image quantizes each color channel into 256 levels. The theoretical data space of the above mentioned image can be expressed by the following format:

$$Q^{W \times H \times C} \tag{1}$$

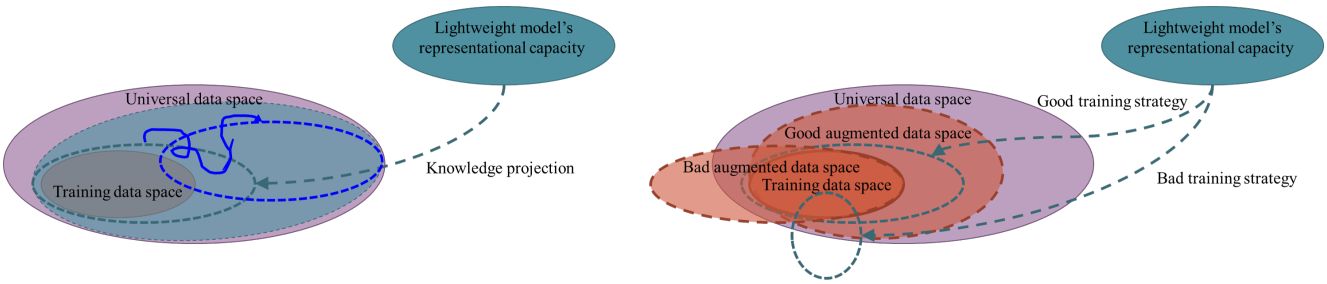
Statistical Data Space: Not all possible combinations in the above theoretical data space will be fully occupied in the real world. In terms of contrast data space, Ojala *et al*. [27] counts the number of occurrences of local binary patterns (LBP) with radius $R = 1$ and number of data points $P = 8$ in multiple texture databases. They found that out of 256 possible patterns, 58 uniform patterns account for 76.6% to 91.8% of the total. In other words, most of the real data are concentrated in a small sub-set in the theoretical data space.

Data Space for Real-World Applications: In the real-world application scenario, we will control the environment of the scene under certain conditions. For example, the control of lighting to make the target area bright, the use of white balance technique or high dynamic ranging technology to enhance the image quality. In terms of semantics, we tolerate the inability to recognize objects that are too far or too small. Therefore, in real-world application scenarios, the universal data space is often smaller than the theoretical data space and the statistical data space.

3.2. Model's representational capacity analysis

In Section 2.1 we introduced two major indicators for estimating the representational capacity of a model, which are the number of regions that the model can segment the space, and the trainability of each computing unit in the model. However, for judging how accurate a model can be on a certain task, one should additionally consider the data attribute and distribution for that type of task.

In the analysis of representational capacity of model, one can easily find that each neuron divides the space into two regions, i.e., active and inactive. This division happens to coincide with the concept of calculating contrast. In other words, they are all active when the output is greater than a certain threshold, otherwise they are inactive. Therefore, we combine the statistical results in [27] with CNN's convolutional kernel. Through this action, we know that 3×3 and 5×5 LBP uniform patterns can represent 89.7% and 54.0% of the data, respectively. The above conclusions bring same useful advantages, that is, even if the trainability of a computing unit is low, it can still represent most of the data through learning a small number of patterns. Therefore, when one estimates the representational capacity of a model, one should calculate the joint expected value of the trainability and the representative data distribution.



(a) Exploring solution space **(b) Exploring augmented data space and training strategy**

Figure 4. Exporing the power of lightweight models. (a) exploring distribution of multiple solution space and (b) exploring suitable augmented data space and training strategy for lightweight models.

3.3. Knowledge projection space analysis

From Figure 3 (a), we see that the optimal knowledge projection space of an over-parameterized huge model covers the entire universal data space, and there are infinitely many sets of solutions. On the other hand, in Figure 3 (b) the best knowledge projection space of an under-parameterized lightweight model also has infinite solutions, but they all fall in the universal data space. Therefore, we must adopt different strategies when optimizing huge models and lightweight models. For huge models, we allow the model's representational range to exceed the data space, and therefore the direction of optimization is to find the maximum intersection between the knowledge projection space and the data space. As for lightweight models, the optimization process must pay attention to the intersection of union (IoU) between the knowledge projection space and the data space. Doing so can avoid wasting the representational capacity of model in the non-existent area of the real data space.

4. Exploring the power of model

After analyzing data space, model's representational capacity, and knowledge projection space, we can then maximally utilize the mutual information between model and data in the knowledge projection space. In Section 4.1, we design a learnable knowledge translation module to explore the multiple solution distribution of the solution space, as shown in Figure 4 (a). In Section 4.2 and 4.3, we will introduce how to combine a genetic algorithm and a grid search-based hyper-parameter search technique to obtain a good data augmentation method and learning strategy, as illustrated in Figure 4 (b). Finally, we will combine the above methods to explore for a suitable knowledge distillation space in Section 4.4.

4.1. Learnable knowledge space translation

We use knowledge space translation to explore the distribution of the infinite set of optimal solutions of a lightweight model in its knowledge projection space. However, if the exploration is done directly in an explicit manner, the multi-layer kernel operation between data and knowledge projection space will have the chain rule effect of updating weights, and it is difficult for the network to converge to a stable state. We proposed an implicit learnable knowledge space translation method based on an implicit kernel space alignment method [36]. Figure 5 illustrates the architecture of the proposed learnable knowledge space translation module.

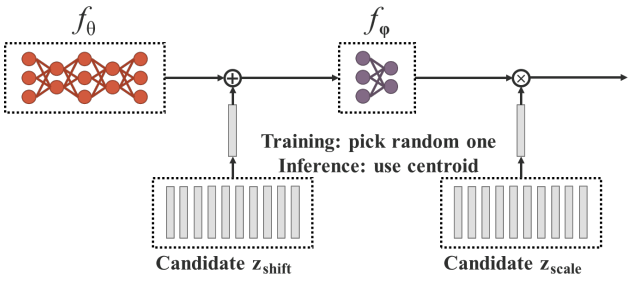


Figure 5. Learnable knowledge space translation module.

In the learnable knowledge translation module, we use addition and multiplication, respectively, to translate and scale up/down in the knowledge projection space. During training, we randomly choose one representative z^i_{shift} and z^j_{scale} from a set Z_{shift} of implicit translation candidate representations and another set Z_{scale} of implicit scaling candidate representations. These two chosen representations are used to obtain the knowledge projection space of the model. Empirically any combination of z^i_{shift} and z^j_{scale} can cover the equal-size space in the solution space. During inferencing, we use the estimated multiple solution distribution centers $mean(Z_{shift})$ and $mean(Z_{scale})$ for knowledge space translation.

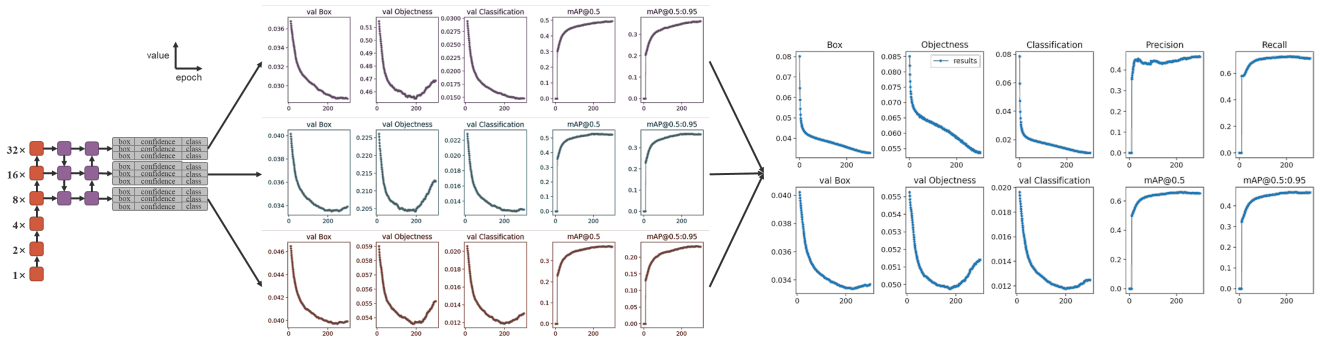


Figure 6. Local (middle) and global (right) loss through training epochs. Middle part shows loss and accuracy of different feature pyramids. Over-fitting starts at different epochs for different feature pyramids. This means we should re-balance the Lagrange multiplier of different feature pyramids. Right part shows the loss of bounding box regression, objectness, and classification starts over-fitting at different epochs. This means we should re-weight the loss of these three objectives.

4.2. Exploring suitable augmented data space

In terms of exploring the augmented data space, we follow YOLOv4 [1] to adopt the genetic algorithm implemented by Glenn *et al*. [16]. We iteratively search the optimal hyper-parameters for data augmentation in the randomly sampled population. We record the results of the update and use it to obtain updated trends. Finally, we generate small grid with the updated trend direction, and then perform grid search to determine the final hyper-parameters needed for data augmentation. Figure 7 shows the process of the proposed hyper-parameter search method. Compared with a pure genetic algorithm, our method is more effective in finding the optimal augmented data space. Compared with the grid search method, we constantly update and provide a better search direction to effectively reduce the number of grid samples.

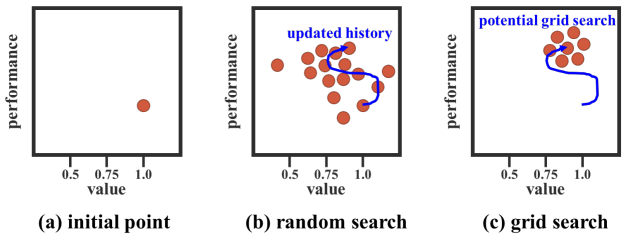


Figure 7. Hyper-parameter search steps.

4.3. Exploring model training strategy

The exploration of model training strategy is almost the same as that of exploring augmented data space. The main difference is that when we generate grid sampling points, we additionally consider both local loss and global loss of the model in each branch output layer, as shown in Figure 6. Basically, the local loss can be used to sample the Lagrangian multiplier of each branch weight update, while the global loss is used to sample the overall training hyper-parameters.

4.4. Exploring suitable knowledge distillation space

The lightweight model has no sufficient representational capacity to distill the entire knowledge from the optimal knowledge projection space of the teacher model. Therefore, when performing knowledge distillation, we must consider both the representational capacity of the target model and the scale of the knowledge projection space learned by the teacher model. Therefore, we will not directly apply the same training strategy and data augmentation method for training the teacher model and distilling knowledge to the lightweight model.

In the previous section, we discussed how to use the lightweight model's representational capacity to get the best knowledge projection space after training. That is to say, the hyper-parameters of the augmented data space and training strategy found by the above method can guide the lightweight model to the optimal knowledge projection space. Therefore, if we use the same set of hyper-parameters to do knowledge distillation, the knowledge projection space obtained by the teacher model should be an over-complete space of the optimal knowledge projection space for the lightweight model. Here, we use this strategy to find a suitable knowledge distillation space.

5. Experiments

The experiment part is divided into six subsections. We will introduce the experiment settings in Section 5.1. Section 5.2 presents training results of the target model and the teacher model under different settings. In Section 5.3, we show the knowledge distillation results of the target model and the teacher model under different settings. The complete ablation study of the three proposed procedures, including joint hyper-parameter search (JHS), suitable knowledge distillation (SKD), and kernel space translation (KST), will be detailed in Section 5.4. In Section 5.5, we will analyze the learnable knowledge space translation. The experiments and applications deployed on low power devices will be shown in Section 5.6.

5.1. Experiment setup

We use MSCOCO dataset [21] to verify the effectiveness of the proposed method. Scaled-YOLOv4 [35] is the state-of-the-art real-time object detector, and it can be scaled up and down to provide excellent object detection function. Basically, it is suitable for low power applications and the study of knowledge distillation, so we choose YOLOv4s as our target model and YOLOv4l as the teacher model. Both preset hyper-parameters and training schedule follow the original Scaled-YOLOv4 settings. Jetson Xavier NX embedded module and Kneron KL720 edge AI SoC are the low power devices used in our system. Since Kneron KL720 edge AI SoC does not support Mish activation function [25], the Mish activation function used in all models in the experiments is replaced by the SiLU activation function [5].

5.2. Target model and teacher model

First, we train the target model (YOLOv4s) and the teacher model (YOLOv4l), and use the results of YOLOv4s as the baseline for the experiment. We use the proposed "exploring suitable augmented data space" method and the "exploring model training strategy" method to perform joint hyper-parameter search (JHS) on YOLOv4s and YOLOv4l, respectively. The hyper-parameter set obtained here are JHS_s and JHS_l, respectively. We use JHS_l to train YOLOv4l to obtain the final teacher model for knowledge distillation, and the experiment results are given in Table 1.

Table 1. Target model and teacher model.

Model	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
A: YOLOv4s [35]	40.2%	59.3%	43.7%	22.9%	45.2%	52.6%
B: YOLOv4l [35]	47.9%	66.9%	51.9%	30.8%	53.2%	61.9%
C: YOLOv4l+JHS_l	**49.0%**	**67.7%**	**53.4%**	**32.7%**	**54.4%**	**63.0%**

* JHS_l means joint hyper-parameter search (JHS) with YOLOv4l.

5.3. Knowledge distillation

In the knowledge distillation experiment, we use the trained teacher model to perform knowledge distillation on YOLOv4s with JHS_s and JHS_l, respectively, and the experiment results are shown in Table 2. The results confirm that the knowledge distillation space does get better results when it is close to the representational capacity of a lightweight model. Our experiment uses JHS_s for distillation to obtain better results, because the final experiment obtained 40.8% AP. It is 0.4% better in AP the experiment using JHS_l for distillation. In other words, the augmented data space and the training strategy we searched with the target model indeed form a suitable knowledge distillation (SKD) space.

Table 2. Knowledge distillation.

Teacher	hyper-p.	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
no	init	40.2%	59.3%	43.7%	22.9%	45.2%	52.6%
C	**JHS_s**	**40.8%**	**59.8%**	**44.1%**	24.0%	**46.4%**	**52.8%**
C	**JHS_l**	40.4%	58.8%	43.7%	**24.2%**	45.6%	51.6%

5.4. Ablation study

After selecting the knowledge distillation model based on the proposed JHS and SKD methods, we add the proposed learnable kernel space translation (KST) module to perform a complete ablation study, and the experiment results are shown in Table 3. In this experiment, the number of candidate shift and scale latent vectors of KST are both set to 3. From the experiment results, it can be found that the three methods we proposed all bring significant improvements to AP. Compared with the baseline, the model we finally trained has a significant increase in AP by 1.2%. This demostrates that our method can effectively explore the power of a lightweight model.

Table 3. Ablation study.

Model	AP^{val}	AP^{val}_{50}	AP^{val}_{75}	AP^{val}_{S}	AP^{val}_{M}	AP^{val}_{L}
A: YOLOv4s [35]	40.2%	59.3%	43.7%	22.9%	45.2%	52.6%
D: A+KST	40.4%	59.3%	43.9%	23.8%	45.5%	53.3%
E: D+JHS	40.6%	59.6%	43.9%	24.1%	45.4%	52.9%
F: E+SKD	**41.4%**	**60.4%**	**44.9%**	**24.2%**	**46.4%**	**53.8%**

* KST: kernel space translation.
* JHS: joint hyper-parameter search.
* SKD: suitable knowledge distillation.

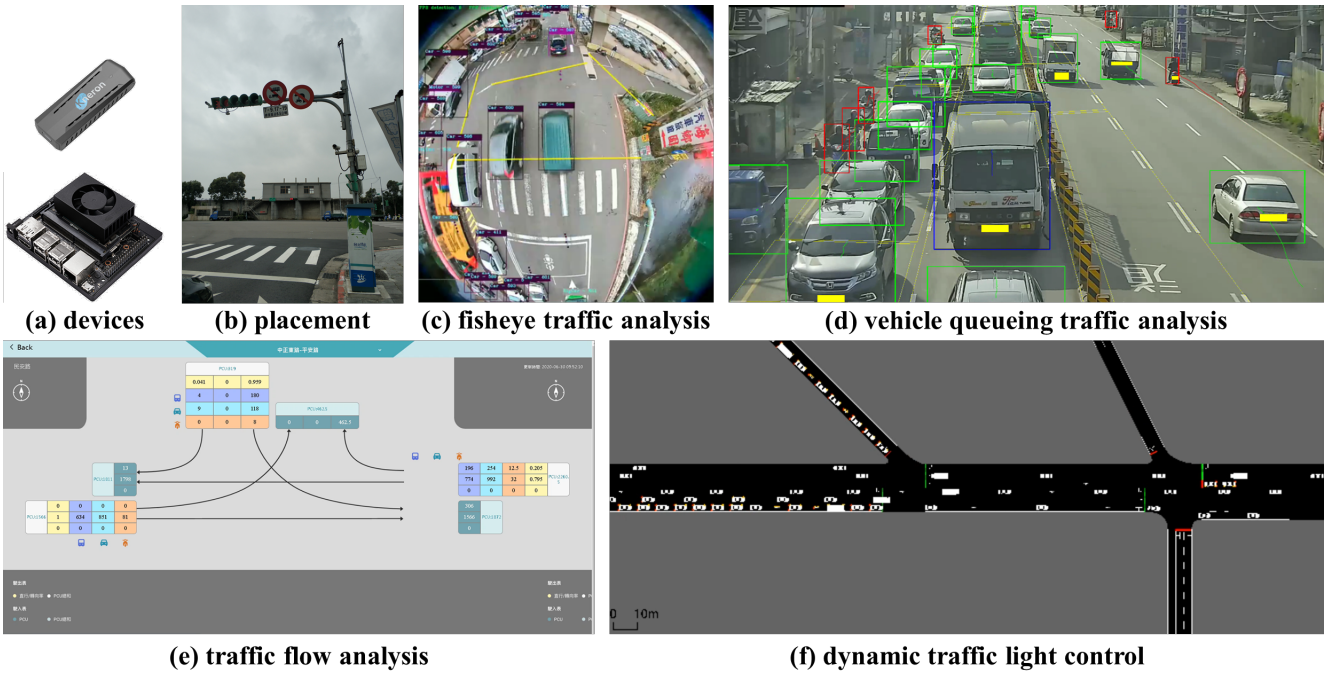


Figure 8. Systems and applications.

5.5. Analysis of knowledge space translation

Table 4 shows the results of all combination of candidate implicit representations in learnable knowledge space translation. The results prove that the proposed method can indeed effectively find multiple approximate solutions in the solution set.

Table 4. Analysis of knowledge space translation.

$\mathbf{ID}_{shift}$	$\mathbf{ID}_{scale}$	$\mathbf{AP}^{val}$	$\mathbf{AP}^{val}_{50}$	$\mathbf{AP}^{val}_{75}$	$\mathbf{AP}^{val}_{S}$	$\mathbf{AP}^{val}_{M}$	$\mathbf{AP}^{val}_{L}$
centroid	centroid	40.4%	59.3%	43.9%	23.8%	45.5%	53.3%
0	0	40.4%	59.3%	43.9%	23.8%	45.5%	53.3%
0	1	40.4%	59.2%	43.9%	23.8%	45.5%	53.3%
0	2	40.4%	59.2%	43.9%	23.8%	45.4%	53.3%
1	0	40.4%	59.3%	43.9%	23.8%	45.4%	53.3%
1	1	40.4%	59.3%	43.9%	23.8%	45.4%	53.3%
1	2	40.4%	59.2%	43.9%	23.8%	45.4%	53.3%
2	0	40.4%	59.3%	43.9%	23.8%	45.5%	53.3%
2	1	40.4%	59.3%	43.9%	23.8%	45.5%	53.3%
2	2	40.4%	59.3%	43.9%	23.8%	45.5%	53.3%

* ID_{shift} and ID_{scale} are index of Z_{shift} and Z_{scale}, respectively.

Table 5. YOLOv4s+KST+JHS+SKD on edge systems

Device	power	Size	FPS	#param.	FLOPs
Kneron KL720	1.725W	416	35	9.3M	10.0G
Nvidia Jetson Xavier NX	15W	640	65	9.3M	21.3G

5.6. Systems and Applications

Finally, we deploy the trained model on the target devices, and the inference speed is listed in Table 5. All data shown in Table 5 can satisfy the low power and real-time requirement. Therefore, we can use the remaining resources to integrate object detection results and apply them to object tracking, traffic analysis, abnormal event analysis, etc. Figure 8 shows several demo of our systems.

6. Conclusions

We have proposed an automated machine learning method that maximizes the performance of a lightweight neural network. Our insight gained from the three main procedures of the proposed method, including learnable knowledge translation module, joint hyper-parameter search for data augmentation and training strategy, and exploring suitable knowledge projection space, can effectively explore the abilities that lightweight models fail to perform in general training process. In MS COCO object detection, our proposed method greatly improves a SOTA lightweight model by 1.2% AP. We have deployed the trained lightweight model in several real application systems, and the performance is satisfactory and robust.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. YOLOv4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 6

[2] Guobin Chen, Wongun Choi, Xiang Yu, Tony Han, and Manmohan Chandraker. Learning efficient object detection models with knowledge distillation. *Advances in neural information processing systems (NeurIPS)*, 30, 2017. 3

[3] Wuyang Chen, Xinyu Gong, and Zhangyang Wang. Neural architecture search on ImageNet in four GPU hours: A theoretically inspired perspective. In *International Conference on Learning Representations (ICLR)*, 2021. 3

[4] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation strategies from data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 113–123, 2019. 3

[5] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018. 7

[6] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. ImageNet-trained CNNs are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations (ICLR)*, 2019. 3

[7] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D Cubuk, Quoc V Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2918–2928, 2021. 3

[8] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. 2

[9] Suyog Gupta and Mingxing Tan. EfficientNet-EdgeTPU: Creating accelerator-optimized neural networks with AutoML. *Google AI Blog*, 2, 2019. 1

[10] Song Han, Huizi Mao, and William J Dally. Deep compression: Compressing deep neural networks with pruning, trained quantization and Huffman coding. In *International Conference on Learning Representations (ICLR)*, 2016. 1, 3

[11] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. In *Workshop on Advances in neural information processing systems (NeurIPS Workshop)*, 2015. 3

[12] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for MobileNetV3. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 1314–1324, 2019. 1

[13] Sheng-Wei Huang, Che-Tsung Lin, Shu-Ping Chen, Yen-Yi Wu, Po-Hao Hsu, and Shang-Hong Lai. AugGAN: Cross domain adaptation with GAN-based data augmentation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 718–731, 2018. 3

[14] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1501–1510, 2017. 3

[15] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018. 3

[16] Glenn Jocher, Yonghye Kwon, guigarfr, perry0418, Josh Veitch-Michaelis, et al. hyperparameter evolution #392, 2019. 6

[17] Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. 3

[18] Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations (ICLR)*, 2017. 2, 3

[19] Edgar Liberis, Łukasz Dudziak, and Nicholas D Lane. μNAS: Constrained neural architecture search for microcontrollers. In *Proceedings of the 1st Workshop on Machine Learning and Systems*, pages 70–79, 2021. 1

[20] Ji Lin, Wei-Ming Chen, Yujun Lin, John Cohn, Chuang Gan, and Song Han. MCUNet: Tiny deep learning on IoT devices. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 1

[21] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 740–755, 2014. 7

[22] Ning Liu, Xiaolong Ma, Zhiyuan Xu, Yanzhi Wang, Jian Tang, and Jieping Ye. Autocompress: An automatic dnn structured pruning framework for ultra-high compression rates. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 34, pages 4876–4883, 2020. 2, 3

[23] Zhuang Liu, Mingjie Sun, Tinghui Zhou, Gao Huang, and Trevor Darrell. Rethinking the value of network pruning. In *International Conference on Learning Representations (ICLR)*, 2019. 1, 3

[24] Joseph Mellor, Jack Turner, Amos Storkey, and Elliot J. Crowley. Neural architecture search without training. In *International Conference on Machine Learning (ICML)*, 2021. 3

[25] Diganta Misra. Mish: A self regularized non-monotonic neural activation function. *Proceedings of the British Machine Vision Conference (BMVC)*, 2020. 7

[26] Guido Montúfar, Razvan Pascanu, Kyunghyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2014. 3

[27] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern*

analysis and machine intelligence (TPAMI), 24(7):971–987, 2002. 4

[28] Jathushan Rajasegaran, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, and Mubarak Shah. Self-supervised knowledge distillation for few-shot learning. *arXiv preprint arXiv:2006.09785*, 2020. 3

[29] Veit Sandfort, Ke Yan, Perry J Pickhardt, and Ronald M Summers. Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks. *Scientific reports*, 9(1):1–9, 2019. 3

[30] Wenqi Shao, Shitao Tang, Xingang Pan, Ping Tan, Xiaogang Wang, and Ping Luo. Channel equilibrium networks for learning deep representation. In *International Conference on Machine Learning (ICML)*, pages 8645–8654, 2020. 2

[31] Zhiqiang Shen, Zhankui He, and Xiangyang Xue. Meal: Multi-model ensemble via adversarial learning. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*, volume 33, pages 4886–4893, 2019. 3

[32] Zhiqiang Shen and Marios Savvides. Meal v2: Boosting vanilla resnet-50 to 80%+ top-1 accuracy on imagenet without tricks. In *Workshop on Advances in neural information processing systems (NeurIPS Workshop)*, 2020. 3

[33] Samuel Stanton, Pavel Izmailov, Polina Kirichenko, Alexander A Alemi, and Andrew Gordon Wilson. Does knowledge distillation really work? *arXiv preprint arXiv:2106.05945*, 2021. 2

[34] Juanhui Tu, Hong Liu, Fanyang Meng, Mengyuan Liu, and Runwei Ding. Spatial-temporal data augmentation based on LSTM autoencoder network for skeleton-based human action recognition. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pages 3478–3482, 2018. 3

[35] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-YOLOv4: Scaling cross stage partial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13029–13038, 2021. 7

[36] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. You only learn one representation: Unified network for multiple tasks. *arXiv preprint arXiv:2105.04206*, 2021. 5

[37] Tao Wang, Li Yuan, Xiaopeng Zhang, and Jiashi Feng. Distilling object detectors with fine-grained feature imitation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4933–4942, 2019. 3

[38] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in neural information processing systems (NeurIPS)*, volume 29, pages 2074–2082, 2016. 2, 3

[39] Mitchell Wortsman, Maxwell Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. Learning neural network subspaces. In *International Conference on Machine Learning (ICML)*, 2021. 2

[40] Lechao Xiao, Jeffrey Pennington, and Samuel Schoenholz. Disentangling trainability and generalization in deep neural networks. In *International Conference on Machine Learning (ICML)*, pages 10462–10472, 2020. 3

[41] Huan Xiong, Lei Huang, Mengyang Yu, Li Liu, Fan Zhu, and Ling Shao. On the number of linear regions of convolutional neural networks. In *International Conference on Machine Learning (ICML)*, pages 10514–10523, 2020. 3

[42] Guodong Xu, Ziwei Liu, Xiaoxiao Li, and Chen Change Loy. Knowledge distillation meets self-supervision. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 588–604, 2020. 3

[43] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. CutMix: Regularization strategy to train strong classifiers with localizable features. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (CVPR)*, pages 6023–6032, 2019. 3

[44] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. MixUp: Beyond empirical risk minimization. In *International Conference on Learning Representations (ICLR)*, 2018. 3

[45] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision (ICCV)*, pages 2223–2232, 2017. 3

[46] Barret Zoph, Ekin D Cubuk, Golnaz Ghiasi, Tsung-Yi Lin, Jonathon Shlens, and Quoc V Le. Learning data augmentation strategies for object detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 566–583, 2020. 3