

Article

Detecting Logos for Indoor Environmental Perception Using Unsupervised and Few-Shot Learning

Changjiang Yin , Qin Ye , Shaoming Zhang * and Zexin Yang

College of Surveying and Geo-Informatics, Tongji University, Shanghai 200092, China; sika@tongji.edu.cn (C.Y.); yeqin@tongji.edu.cn (Q.Y.); zexinyang@tongji.edu.cn (Z.Y.)

* Correspondence: zhangshaoming@tongji.edu.cn

Abstract: Indoor scenes are crucial components of urban spaces, with logos serving as vital information within these environments. The accurate perception of logos is essential for effectively operating mobile robots in indoor environments, which significantly contributes to many upper-level applications. With the rapid development of neural networks, numerous deep-learning-based object-detection methods have been applied to logo detection. However, most of these methods depend on large labeled datasets. Given the fast-changing nature of logos in indoor scenes, achieving reliable detection performance with either the existing large labeled datasets or a limited number of labeled logos remains challenging. In this article, we propose a method named MobileNetV2-YOLOv4-UP, which integrates unsupervised learning with few-shot learning for logo detection. We develop an autoencoder to obtain latent feature representations of logos by pre-training on a public unlabeled logo dataset. Subsequently, we construct a lightweight logo-detection network and embed the encoder weights as prior information. Training is performed on a small dataset of labeled indoor-scene logos to update the weights of the logo-detection network. Experimental results on the public logo625 dataset and our self-collected LOGO2000 dataset demonstrate that our method outperforms classic object-detection methods, achieving a mean average detection precision of 83.8%. Notably, our unsupervised pre-training strategy (UP) has proven effective, delivering a 15.4% improvement.

Keywords: logo detection; autoencoder; unsupervised learning; few-shot learning



Citation: Yin, C.; Ye, Q.; Zhang, S.; Yang, Z. Detecting Logos for Indoor Environmental Perception Using Unsupervised and Few-Shot Learning. *Electronics* **2024**, *13*, 2246. <https://doi.org/10.3390/electronics13122246>

Academic Editor: Paolo Tripicchio

Received: 12 April 2024

Revised: 5 June 2024

Accepted: 6 June 2024

Published: 7 June 2024



Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The operation of mobile robots in large and complex indoor scenes, integral components of urban spaces, heavily relies on the ability to perceive their environment accurately. Logos serve as prominent markers within indoor environments, and recognizing these logos is an effective remote-sensing technique for environmental perception. This recognition provides vital information about the environment and significantly aids in the positioning of mobile robots. Therefore, these robots need to detect and recognize logos from images captured in commercial indoor scenes, which is beneficial for applications such as object-level simultaneous localization and mapping (SLAM) [1–3] and autonomous navigation [4–6]. However, the vast variety of logos in indoor environments complicates the design of robust handcrafted features. Furthermore, the continuous creation of new logos without a fixed set of categories makes it challenging to create a large, labeled, and comprehensive dataset for deep-learning techniques. Consequently, detecting and recognizing indoor logos remains challenging, severely restricting the environmental perception capabilities of robots in indoor spaces and impacting their applications within such spaces.

Previous methods for logo detection have primarily relied on handcrafted feature-extraction techniques, such as scale-invariant feature transform (SIFT) and histogram of oriented gradients (HOG). However, they are manually designed for features of logos with stable categories and cannot ensure robust output when confronted with constantly emerging and changing logos. Additionally, the sliding window-based selective region

search algorithm employed in these methods leads to a substantial waste of time, especially when dealing with a large variety of logos, further compromising their efficiency. With the advancement of deep learning and neural networks, many logo-detection studies have been carried out based on classic object-detection models. Currently, many deep-learning techniques rely on large labeled datasets. However, acquiring a large labeled dataset that reflects the distribution of logos in real-world indoor scenes is challenging in engineering. This makes it difficult to directly apply these methods.

Recently, unsupervised learning and few-shot learning have played an important role in addressing the difficulties in data annotation and collection. Unsupervised-learning methods have traditionally relied on clustering techniques and dimensionality reduction methods. Recent research has mainly focused on deep-learning methods, which include autoencoders and the restricted Boltzmann machine (RBM) for feature dimensionality reduction, as well as the generative adversarial network (GAN) for data generation. Hitton et al. [7] believe that unsupervised learning can effectively provide prior initialization information, and some recent studies have applied unsupervised learning to neural network pre-training. Algorithm adjustment, model adjustment, and data augmentation are the three main strategies of few-shot learning that enhance the learning capability of the network and alleviate the dependence on traditional large labeled datasets. Although both unsupervised learning and few-shot learning have proven to be beneficial, relying on only one of them may not suffice to overcome the challenges posed by the absence of a large labeled logo dataset. Unsupervised learning alone cannot fully address the problem of limited sample availability, while few-shot learning still requires some prior knowledge of logos that we do not possess yet. Therefore, the combination is necessary for complementarity. However, in the field of object detection, particularly in logo detection, there is a lack of research on the integration of unsupervised learning and few-shot learning.

Based on the above analysis, in this work, we propose a deep-learning-based logo-detection framework that combines few-shot fine-tuning and unsupervised learning. Considering the need for real-time performance and the existing limits of computation in engineering applications, we opt for a lightweight backbone network and a one-stage object-detection architecture. We utilize the autoencoder to carry out unsupervised pre-training (UP) on a large unlabeled logo dataset. This allows us to acquire valuable prior knowledge of logos, which we can then transfer to the logo-detection network for efficient few-shot fine-tuning. In our method, the UP technique is constant for all indoor scenes and needs to be performed only once. If logos change or new categories of logos are introduced, only a few-shot fine-tuning is required for updates. We conducted experiments on a logo dataset based on self-collected indoor-scene images. We compared the results with three baseline methods and demonstrated the effectiveness of our approach. In particular, we confirmed the efficiency of our UP technique through ablation experiments.

Our main contributions are summarized as follows.

- (1) We propose an autoencoder that combines MobileNetV2 and U-Net for logo detection pre-training. Compared to traditional autoencoders, the deeper network structure of MobileNetV2 and the introduction of depth-wise separable convolution enhance the representation of logos;
- (2) We integrate MobileNetV2 with YOLOv4 for logo detection to achieve transfer learning. The introduction of a lightweight backbone not only improves the speed of logo detection but also enhances precision;
- (3) We propose a deep-learning framework that combines few-shot and unsupervised learning, which overcomes the dependence of traditional deep-learning methods on large labeled logo datasets. Additionally, we avoid a second massive training when new logos emerge, and only a small dataset of labeled indoor-scene logos is required for fine-tuning.

2. Related Work

2.1. Logo Detection

Logo detection is a specific application of object detection. The task involves identifying the location of the logo and determining its category. Therefore, logo-detection methods share similarities with other object-detection methods. Traditionally, logo detection relied on hand-crafted features, like SIFT [8–10] and HOG [11,12], as well as sliding window-based localization, and machine-learning classifiers, such as support vector machine (SVM), before the advent of rapid neural network and deep-learning development. However, these traditional methods suffer from inherent limitations in terms of real-time performance and precision, primarily stemming from their limited logo representation capabilities. In recent years, logo-detection research has primarily focused on deep learning due to the significant advancements in feature extraction and object detection. These methods include region-based convolutional neural network (R-CNN) models, you only look once-based (YOLO) models, single-shot detector-based (SSD) models, and other models.

Hoi et al. [13] first introduced deep-learning methods to logo detection by using selective search [14] in R-CNN to generate regions of interest (ROI). Li et al. [15] employed transfer learning, data augmentation, and clustering to build a logo-detection system based on Fast R-CNN [16]. Chen et al. [17] improved the detection of small logos using Faster R-CNN [18] with residual structures. Yin et al. [19] utilized a logo-detection network based on YOLOv2 [20] for vehicle logo recognition, while Yang et al. [21] proposed a data-driven enhanced training method based on YOLOv3 [22] for vehicle logo recognition. Wang et al. [23] introduced a large logo dataset with diverse categories and improved loss functions in YOLOv3. Jiang et al. [24] integrated deformable convolutional residuals and convolutional transformers into YOLOv4 [25] to mitigate background interference in logo detection. Orti et al. [26] developed a logo-detection method based on SSD [27] and InceptionV2, while Zhang [28] proposed a vehicle logo-extraction method based on SSD and adjusted the pre-training strategy to adapt to logos of multiple scales. In addition to the three main methods (R-CNN, YOLOs, and SSD) mentioned above, some other models have also been used for logo detection, such as feature pyramid networks (FPN) [29] and AttentionMask [30]. Li et al. [31] incorporated a new operator and self-attention mechanism into the convolutional neural network (CNN) to detect logos of different scales and aspect ratios. Jain et al. [32] designed a spatial attention module to improve logo-detection performance. Yang et al. [33] proposed a transformer-based network to enhance the representation capability of logos. In the above scenarios that did not consider indoor-scene diversity, the variety of logos is limited and cannot accommodate the appearance of the new logo. We constructed a YOLOv4-based network utilizing the MobileNetV2 backbone network for logo recognition in indoor images. We chose YOLOv4, considering the small sizes of logos in indoor images, as well as the need for real-time logo detection.

2.2. Few-Shot Learning

Currently, a large number of deep-learning methods for logo detection still rely on supervised learning with large labeled datasets, such as LogoDet-3K [23] and QMUL-OpenLogo [34]. However, many engineering applications, including logo detection, face difficulties in acquiring labeled data, thus elevating the significance of few-shot learning. In [35], a method based on the backbone pre-training model by ImageNet and fine-tuning it on a few labeled samples was proposed, and Chen et al. [36] first introduced the few-shot fine-tuning strategy to the field of object detection. Meta learning has been introduced to enhance the capability of few-shot learning. Mandic et al. [37] proposed a gradient descent strategy for learning both neural network parameters and meta-learning parameters on a few labeled samples. Fei et al. [38] applied meta learning to the training and updating of network optimizers to adapt to few-shot learning scenarios. Another few-shot learning strategy is based on model adjustment. Yan et al. [39] utilized convolutional neural networks to learn the compression parameters of samples and embedded a few samples into a low-dimensional space for model optimization. In [40], an external memory module was

constructed to store prior information, and joint encoding with a few labeled samples was performed to enhance the sample-representation capability. Martin et al. [41] proposed to generate new data with a few labeled samples based on probability distribution models. Furthermore, data augmentation is also commonly employed in few-shot learning. Li et al. [42] explored the potential of GAN by using similar datasets to generate data with the same distribution as a few labeled samples. He et al. [43] expanded the dataset by learning a one-to-many transformation model on a few labeled samples. In [44], an SVM classifier learned on a few labeled samples was utilized to provide class labels for weakly labeled data, forming a larger labeled dataset. Due to the absence of an existing few-shot learning method specifically designed for logos in indoor scenes, our research will explore the potential of fine-tuning in logo detection.

2.3. Unsupervised Learning

Object detection also suffers from difficulties in data annotation, which is reflected in both time and cost. Therefore, unsupervised learning or self-supervised learning has gradually gained popularity among researchers. Traditional unsupervised methods include clustering and dimensionality reduction techniques. K-means [45] and the balanced iterative reducing and clustering using hierarchies (BRICH) algorithm [46] are classic clustering methods. The density-based spatial clustering of applications with noise (DBSCAN) method [47], proposed by Ester et al., is a density-based clustering approach and derives many variants [48,49]. Principal component analysis (PCA) [50] is a classic linear dimensionality reduction method, while singular value decomposition (SVD) [51] is also an effective linear technique. Other unsupervised dimensionality reduction methods, such as independent component analysis (ICA) [52] and t-distributed stochastic neighbor embedding (t-SNE) [53] utilize non-linear transformations.

In recent years, more unsupervised-learning methods based on deep learning have emerged; Autoencoder, RBM, and GAN are typical. Autoencoder [54] uses a neural network structure encoder to extract latent features from images and reconstruct them based on the latent features, which combines a backpropagation algorithm [55] with self-supervision. Liou et al. used an autoencoder [56] to extract latent information in the unsupervised text. RBM [57], proposed by Paul et al., optimizes the similarity probability between the input and output in a statistical manner. In [58], stacking RBMs was proposed to improve unsupervised learning. GAN [59], a similar recombination of autoencoder structures, trains the data-generation ability of GAN through the game between the generator and the discriminator. Several studies have employed unsupervised learning for network pre-training. Hitton et al. [7] demonstrated the effectiveness of using self-supervised learning to initialize neural network parameters. Unsupervised-learning methods, such as the simple framework for contrastive learning of visual representations (SimCLR) [60] and momentum contrast (MoCo) [61], have been proposed as alternatives to ImageNet [62] for pre-training the backbone network. In the works of Bar et al. [63] and Wang et al. [64], the backbone network is jointly pre-trained with the detector in an unsupervised manner.

In this work, we deeply explore the capability of autoencoders in logo latent feature extraction and propose a UP strategy based on autoencoders.

3. Proposed Method

In this section, we provide a comprehensive description of our proposed MobileNetV2-YOLOv4-UP. First, in Section 3.1, we show the framework of MobileNetV2-YOLOv4-UP along with its significant components, which provide an overall understanding of our method. Then in Section 3.2, we provide a detailed description of MobileNetV2-AE. This component plays a pivotal role in learning representations of logo features, concurrently serving as the primary means for the UP of logo data in our method. Afterward in Section 3.3, we explain the architecture of the logo-detection network based on a lightweight backbone, which is an important part of few-shot fine-tuning. Finally, in Section 3.4, we detail how to combine UP with few-shot learning. We elucidate the process of transferring

the knowledge obtained from UP based on an autoencoder to the object-detection network, thereby enhancing the effectiveness of few-shot learning.

3.1. Overall Framework

The framework of our proposed MobileNetV2-YOLOv4-UP is shown in Figure 1. MobileNetV2-YOLOv4-UP includes three main components: an unsupervised-learning module for UP, a few-shot learning module, and a logo-detection module. First, unsupervised learning plays the role of backbone pre-training and is conducted on a meticulously designed MobileNetV2-AE autoencoder. A large unlabeled logo dataset, logo625, is used to obtain robust logo representations. Then, few-shot learning is conducted in a fine-tuning way. The prior information from UP is transferred to MobileNetV2-YOLOv4, and we perform supervised fine-tuning on a small dataset, LOGO2000, of labeled indoor-scene logos. This step aligns the model with the real scene distribution, addressing differences between logo625 (pure logos) and LOGO2000 (indoor scenes).

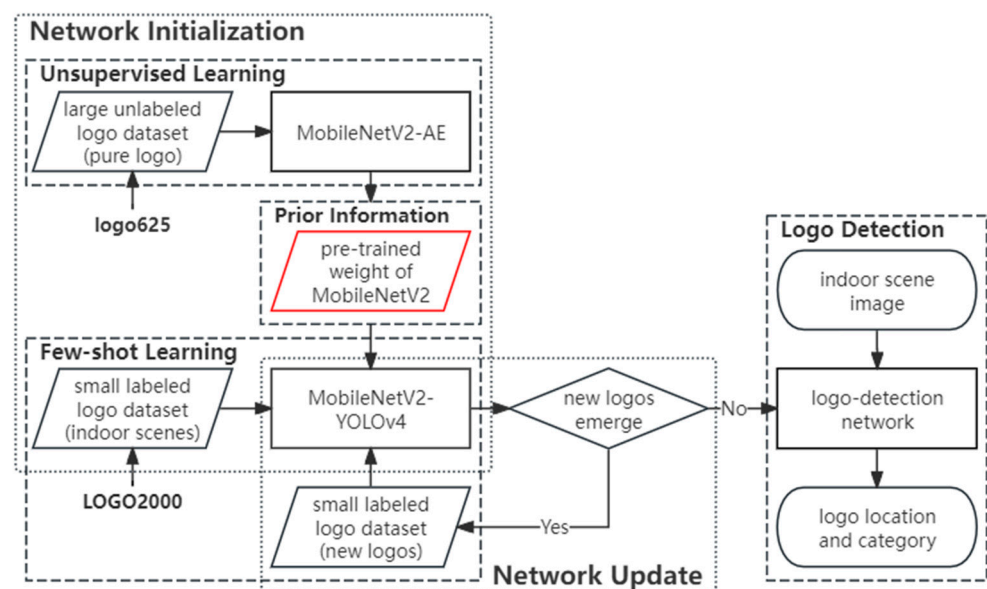


Figure 1. Framework of the proposed MobileNetV2-YOLOv4-UP. First, we performed unsupervised learning to pre-train the backbone network and acquire prior knowledge (red frame). This prior knowledge was then transferred to the logo-detection network, followed by supervised fine-tuning on a small dataset. The unsupervised learning was achieved using the MobileNetV2-AE autoencoder, with its encoder being the same as the backbone network in MobileNetV2-YOLOv4, thus enabling the pre-training of the backbone network.

After the initialization of the logo-detection network, it is essential to determine if the network requires updates. When new logos emerge, a small dataset consisting of labeled new logos is used to fine-tune MobileNetV2-YOLOv4. It should be noted that the update of the logo-detection network is only related to few-shot learning, which means UP only needs to be conducted once in our method. This can be attributed to the robust generalization capabilities of the logo features obtained through UP. Our method can effectively address challenges in data collection and annotation. Additionally, the integration of a lightweight backbone and one-stage object-detection network also reduces computational consumption.

3.2. MobileNetV2-AE-Based Unsupervised Learning

The autoencoder is one of the most widely used unsupervised-learning methods. Although it is traditionally used for data reconstruction, it also performs well in feature learning. Autoencoder serves as the core module for UP in our method, which plays the role of learner of robust logo features. The autoencoder pre-training can well provide prior information for logo-detection networks, so we only need to fine-tune a few images.

Traditional autoencoders are mostly designed based on CNN or multi-layer perceptron (MLP) with a small number of layers, making it difficult to achieve a good representation of the relatively complex logos. Another problem with using these autoencoders for UP is the difficulty in transferring the learned prior knowledge to the object-detection network because there are often large differences in network structure between autoencoders and object-detection networks. Since the encoder is used to implement feature extraction in the autoencoder, a good idea to solve the above problem is to choose the same network both for the encoder and the backbone of the object-detection network.

Since our logo detection is used for environmental perception, it puts certain requirements on timeliness. In addition, our UP is carried out on a large dataset, which is a big challenge for autoencoder training. In consideration of the above issues, in this work, we employed a lightweight MobileNetV2 [65] as the encoder structure, given its use of inverted residual structures and its outstanding performance in classification and detection tasks. It will also be helpful in terms of detection speed and training time.

A typical MobileNetV2 network has 154 layers, but we only selected the first 152 layers of the MobileNetV2 network. As the subsequent two layers are used to reduce the channel and size of the feature map, respectively, they play little role in feature extraction. After being encoded by MobileNetV2, the size of the feature map is reduced by a factor of 32. Consequently, the size of the input image X_i must be a multiple of 32. To balance training time and effectiveness, we ultimately set the input size of the training data to 224×224 in our study. Additionally, we aim to further enhance the representational capacity of the latent feature that owns semantic attributes. To achieve this, we first conduct feature dimension reduction. We flatten the output F_m encoded by MobileNetV2 (Enc_{mv2}) from a $7 \times 7 \times 320$ tensor to a $15,680 \times 1$ vector F_f , and then obtain the 1024-dimensional latent feature F_l through an MLP layer. Second, we add L2 regularization to the kernel in the MLP layer to prevent overfitting on dataset logo625 and enhance the encoder's latent features' generalizability. The formula for L2 regularization is shown in Equation (1):

$$\min < \left[\frac{1}{N} \sum_{i=1}^N (y_i - \omega^T x_i)^2 + C \|\omega\|_2^2 \right] > \quad (1)$$

$$\|\omega\|_2^2 = \sqrt{\sum_{i=1}^n \omega_i}$$

Here, N represents the number of samples used for training. C is a manually defined constant and is set to 0.01 in this work. x_i and y_i are the input and output of the neural network layer respectively, and ω is the weight of this layer. ω consists of n vectors ω_i . The encoding process is illustrated in Equation (2).

$$\begin{aligned} F_m &= Enc_{mv2}(X_i) \\ F_f &= Flatten(F_m) \\ F_l &= MLP_{kernel-L2}(F_f) \end{aligned} \quad (2)$$

For the decoder part, we constructed a decoder network based on U-Net [66]. U-Net consists of two parts, the feature-extraction part, and the reconstruction part, while the reconstruction part has a similar structure to the decoder in autoencoder. U-Net is capable of handling complex segmentation tasks, making its reconstruction part a highly feasible choice for our decoder. Referring to the design of the last two layers in the encoder, we made a symmetrical design in the decoder. We first increase the dimension of the latent feature F_l from 1024 to 15,680 and get vector F_u through an MLP layer, then reshape F_u into a $7 \times 7 \times 320$ tensor $\tilde{F}_r$. Finally, we input $\tilde{F}_r$ into the reconstruction part of U-Net (Dec_{unet}), and the output $\tilde{X}_i$ is the reconstructed image. An important aspect to mention is that U-Net generally incorporates multi-scale features to achieve better reconstruction results. However, since we want to enhance the encoding capabilities of the encoder, we hope that the reconstruction of the autoencoder is completely based on latent features., it

requires that the high-level latent features must own the strongest representation ability of the input. Therefore, in the autoencoder model we constructed, we removed the multi-scale feature connections so that the decoder only reconstructs based on the last features encoded by the encoder, in order to enhance its encoding ability. The decoding process is illustrated in Equation (3).

$$\begin{aligned} F_u &= MLP(F_l) \\ F_r &= Reshape(F_u) \\ \tilde{X}_i &= Dec_{unet}(F_r) \end{aligned} \quad (3)$$

Our MobileNetV2 autoencoder structure is shown in Figure 2, where the input image is first resized to 224×224 and then encoded by the MobileNetV2 to obtain a 1024-dimensional latent feature. The latent feature vector is then input into the decoder to reconstruct the image, and the encoder's capability to extract logo latent features is improved by minimizing the difference between the reconstructed image $\tilde{X}_i$ and the original image X_i (Equation (4)).

$$\operatorname{argmin} < \left[\Delta \left(X_i, \tilde{X}_i \right) \right] > \quad (4)$$

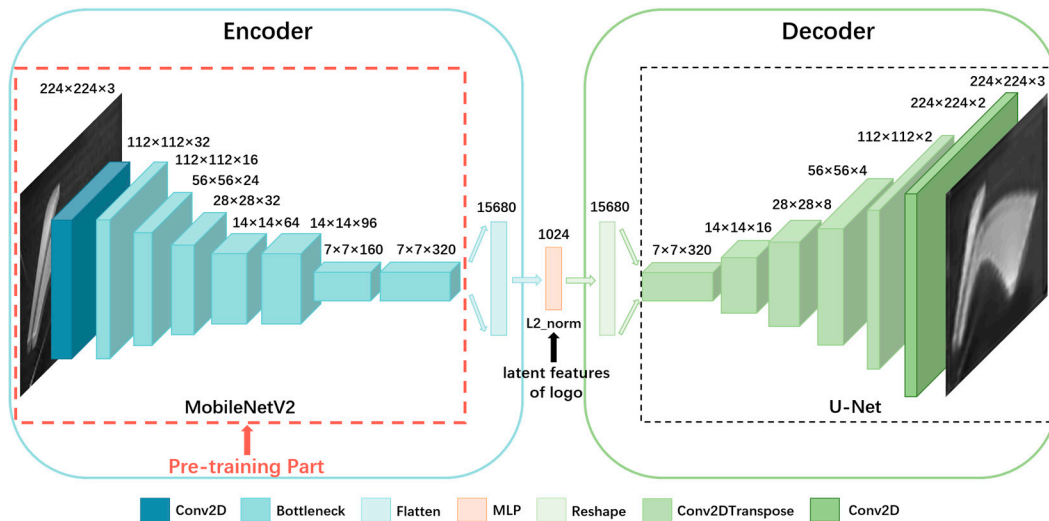


Figure 2. Architecture of MobileNetV2-AE. Unsupervised learning is conducted on MobileNetV2-AE, aiming to pre-train the encoder, MobileNetV2, to enhance its ability to extract robust logo representations.

In terms of loss-function selection, binary cross entropy (BCE) is more suitable for recognition tasks (including object detection) than mean squared error (MSE) [67]. Using BCE instead of MSE in pre-training the autoencoder facilitates better alignment with our detection network, ensuring the learned representations are more suitable for recognition tasks. While MSE is commonly used in autoencoder training, its limitation in capturing categorical features crucial for detection makes BCE a more appropriate selection. Therefore, we choose BCE as the loss function for autoencoder training. Let M denote the number of samples, and the input X_i consists of n vectors $x_{j,i}$. The loss function can be expressed as:

$$L_{CE} = -\frac{1}{M} \sum_{i=1}^M \sum_{j=1}^n \left[x_{j,i} \log \tilde{x}_{j,i} + (1 - x_{j,i}) \log (1 - \tilde{x}_{j,i}) \right] \quad (5)$$

By pre-training the autoencoder on a large unlabeled dataset, the optimal latent feature representation for logos can be obtained. Compared to large labeled datasets, a large unlabeled dataset is easier to acquire. The autoencoder does not require a second training, reducing system maintenance complexity. As pointed out in [68], the effectiveness of fine-

tuning is influenced by the correlation between the original task used for pre-training and the new task. The datasets used for the two tasks are supposed to share certain similarities, and the downstream task is logo detection. Therefore, we need to acquire a large unlabeled logo dataset for autoencoder pre-training. Currently, there is a wide availability of public datasets (e.g., logo625), and acquiring data through web crawling is also easy.

3.3. MobileNetV2-YOLOv4-Based Logo Detection

Currently, object-detection methods based on deep learning can be roughly divided into two categories: one-stage object-detection methods and two-stage object-detection methods. Although two-stage methods may be superior in detection accuracy, they usually involve more computational steps, such as region proposal generation, feature extraction, classification, and bounding box regression. The one-stage method directly detects the target in the image in a single inference stage, with fewer processing steps, so the inference time is shorter. Since our scenario requires real-time detection performance, a one-stage approach is more suitable.

The YOLO series [20,22,25] is the most widely used one-stage object-detection method. In the real indoor-scene images we collected, logos often occupy a relatively small proportion within the image. YOLOv4 incorporates an FPN structure, which is effective for detecting objects at various scales. FPN generates feature maps at different resolutions, allowing the model to detect small objects by leveraging features from multiple scales. It is expected that YOLOv4 performs better at detecting small-sized logos. Therefore, we adopted a network based on YOLOv4 as the logo-detection framework. As mentioned in 3.2, to transfer the prior knowledge obtained from UP learning, the backbone of the object-detection network should be consistent with the encoder. We replaced the original CSPDarknet53 backbone network used in the YOLOv4 with MobileNetV2 due to the ease of pre-training weight transfer from UP. Furthermore, MobileNetV2 serves as a lightweight backbone network compared to CSPDarknet53, aligning with our requirement for real-time detection efficiency. Due to the introduction of depthwise separable convolution in MobileNetV2, the convolution operations and network parameters can be significantly reduced, thereby reducing computational consumption.

Inspired by the design in [69], our MobileNetV2-YOLOv4 architecture is similar to YOLOv4, consisting of three parts: the backbone network, spatial pyramid pooling (SPP), and path aggregation network (PANet). SPP enhances YOLOv4 by extracting multi-scale features through different pooling operations, which increases the model's receptive field and its ability to detect objects of varying sizes. PANet improves feature fusion by efficiently aggregating features from different layers, enhancing context information and overall detection accuracy. In the UP part, we only selected the first 152 layers of the MobileNetV2 to construct the encoder in MobileNetV2-AE. Therefore, we adopted the same approach to building the backbone (Enc_{mv2}) of the logo-detection network. Since PANet adopts a multi-scale feature fusion strategy, we need to specify the output layers of MobileNetV2. Referring to the output of CSPDarknet53, we input the feature F_3 and F_5 , respectively, from the fourth bottleneck layer (bottleneck3) and the sixth bottleneck layer (bottleneck5) of MobileNetV2 into PANet (shown in Figure 3). The process is illustrated in Equation (6).

$$\begin{aligned} F_3, F_5, F_b &= Enc_{mv2}(X_i) \\ F_{mt} &= SPP(F_b) \\ F_{fus} &= PANet(F_3, F_5, F_{mt}) \end{aligned} \quad (6)$$

Here, X_i is the input image of the logo-detection network, and F_b represents the output of the last layer in MobileNetV2. F_{mt} represents the result of multi-scale feature fusion achieved through SPP based on F_b . F_{fus} represents another integration of multi-scale features; these features include F_{mt} from SPP and backbone's outputs F_3 and F_5 . Thus, F_{fus} can adapt to logos of different sizes and is input into YOLO Head to obtain the inference results.

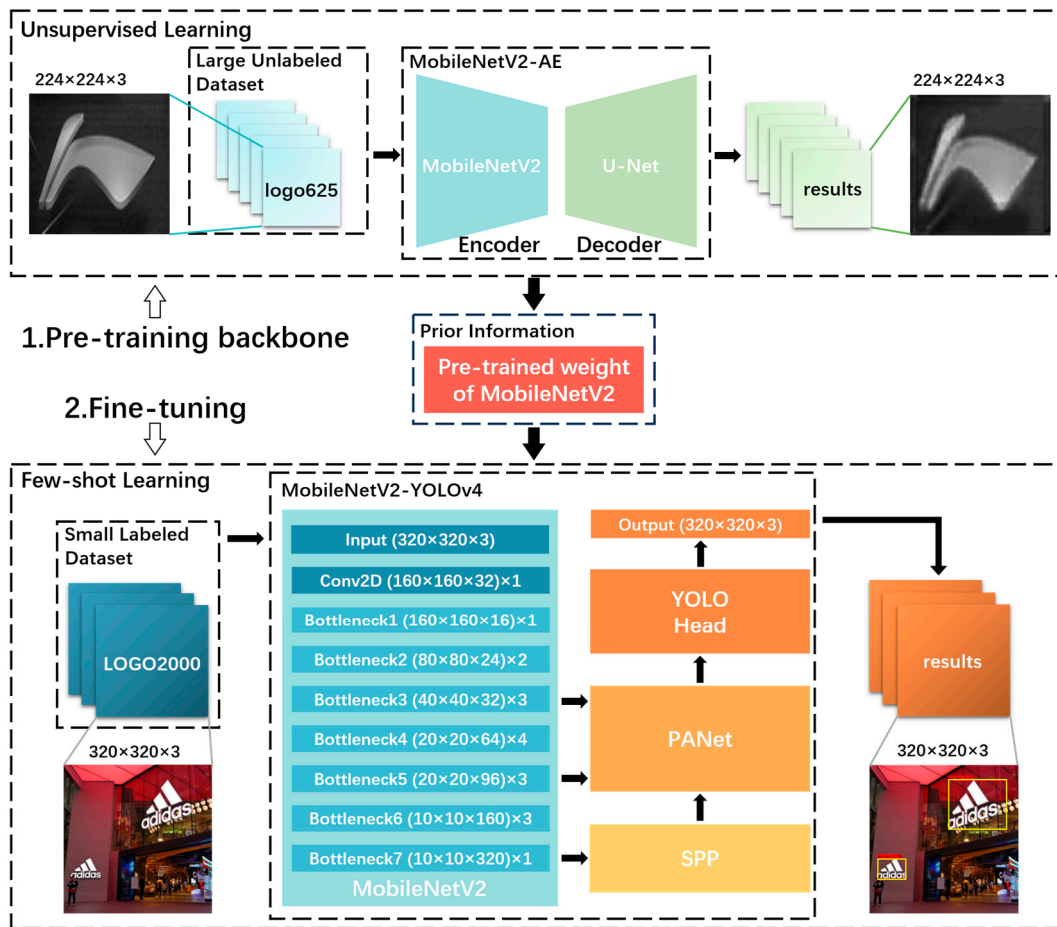


Figure 3. Initialization scheme architecture of MobileNetV2-YOLOv4-UP. Unsupervised learning is first performed to learn the prior knowledge of logos. The prior knowledge learned in the unsupervised-learning part refers to the pre-trained weights of MobileNetV2, which encompass robust representations of logos. This prior knowledge is then transferred to the logo-detection network for initializing the backbone.

In the setting of the loss function, our MobileNetV2-YOLOv4 combines three types of losses: localization loss $Loss_{loc}$ (using complete intersection over union to measure the accuracy of predicted bounding boxes), confidence loss $Loss_{conf}$ (using BCE to measure the model's prediction of object presence and its confidence level), and classification loss $Loss_{cls}$ (using BCE to measure the accuracy of predicted class probabilities). The loss function can be expressed as

$$Loss_{total} = Loss_{loc} + Loss_{conf} + Loss_{cls} \quad (7)$$

To balance training time and detection performance, we set the input size of MobileNetV2 to $320 \times 320 \times 3$ and obtained the bottom-level feature map of size of $10 \times 10 \times 320$. Although the input size of the logo-detection network is larger than the input size of MobileNetV2-AE for UP, there is no impact on the transfer of prior knowledge. Since the dataset used in fine-tuning the logo-detection network is much smaller than the dataset used in UP, it is reasonable to appropriately increase the input size to obtain better detection results without causing a large increase in training time.

3.4. Integration of Unsupervised Learning and Few-Shot Learning for Logo Detection

The initialization scheme architecture of our proposed logo-detection network, as shown in Figure 3, mainly consists of two modules: unsupervised learning and few-shot learning. The network structure of the two modules corresponds to the MobileNetV2-AE autoencoder proposed by us and the YOLOv4 object-detection network based on the

MobileNetV2 backbone. We have already explained in 3.2 that unsupervised learning plays the role of pre-training, so we initially perform UP and then perform few-shot learning. First, we conduct UP on the MobileNetV2-AE autoencoder using a large unlabeled logo dataset without class labels and background noise. Then, we extract the weights of the encoder part in MobileNetV2-AE and transfer them into the MobileNetV2-YOLOv4 object-detection network as the initialization weights of the backbone network. Finally, we fine-tune the MobileNetV2-YOLOv4 object-detection network on a small dataset of labeled logos from large commercial indoor spaces to detect logos. Although the first step still requires training on a large dataset, obtaining a dataset without any labels is more accessible compared to a labeled one.

In addition, we hope that the autoencoder can learn the common features of all logos, which helps to enhance the generalization capability of the encoder. Therefore, in terms of dataset composition, categories of logos in the dataset used for UP can be different from those of logos distributed in indoor scenes. This not only alleviates the difficulty of obtaining specific logo data but also avoids the repetition of UP when the logo's categories change in real indoor scenes. The size of the input image ($320 \times 320 \times 3$) in the logo-detection network is slightly larger than the input size ($224 \times 224 \times 3$) of the autoencoder. We perform this mainly for the following four considerations. 1. A larger input size can retain more details, which improves the detection of small-size logos. 2. The encoder in MobileNetV2-AE shares the same network structure as the backbone in the logo-detection network, and pre-training weight transfer will not be affected. 3. The encoder possesses strong generalization capabilities and can adapt to different input sizes. 4. We balance the training-time cost and computational consumption of UP with the fine-tuning effect of few-shot learning.

4. Experimental Results

4.1. Experimental Protocol

4.1.1. Dataset

Our network consists of two parts: the autoencoder (MobileNetV2-AE) for UP and the logo-detection network (MobileNetV2-YOLOv4). Therefore, in our experiments, the training of both parts is involved. Specifically, we conduct UP of the autoencoder on a large unlabeled dataset logo625 and, then, fine-tune the logo-detection network on a small dataset LOGO2000 of labeled indoor-scene logos.

For the UP part, we used the logo625 dataset (Figure 4), a publicly available dataset developed through web crawling. Logo625 covers a wide range of common brand logos, including 625 categories across various domains, such as apparel, sports, leisure, and food. We used 100,000 images from this dataset for training, approximately 150–180 images per category. We chose this dataset because the logo data in logo625 mostly consists of pure foreground logo images without background information. So, the autoencoder mainly learns the representations of the foreground—the logos. And a small amount of data with background noise would not have a significant impact on our pre-training results. Therefore, pre-training on this dataset can make the encoder more capable of extracting logo features.

Due to the lack of logo datasets for object detection in large indoor scenes, we created a small dataset of labeled indoor-scene logos by web crawling, named LOGO2000 (Figure 5). This dataset includes 20 logo categories, with 100 images per category, for a total of 2000 images. We divided the dataset into training and testing sets at a ratio of 7:3, where 1400 images were used for fine-tuning the logo-detection network, and 600 images were used to evaluate the experimental results. Since the number of logos in each image may differ, the number of objects used for training varies among categories but is generally between 100 and 160.



Figure 4. Example images of logo625.

4.1.2. Platform

All experiments were conducted on a cloud server, with hardware specifications of 30 GB RAM, 14 vCPU Intel(R) Xeon(R) Gold 6330 CPU @ 2.00 GHz, and Nvidia RTX A5000 (24 GB). The software environments used for the experiments were Ubuntu 18.04, TensorFlow 1.15.5, Keras 2.1.5, Python 3.8, and Cuda 11.4.

4.1.3. Evaluation Metrics

We employed and calculated mean average precision (mAP), which is the most commonly used and important metric for evaluating object-detection performance, to assess the effectiveness of our model. The mAP combines the concepts of precision and recall to provide a comprehensive measure of a model's performance. It calculates the average precision (AP) for each class and then takes the mean of these values. AP is defined as the area under the precision–recall curve. Additionally, we evaluated the detection speed of the network on logos by measuring its frames per second (FPS). The calculation formula for mAP is illustrated in Equation (8).

$$\text{mAP} = \frac{1}{N} \sum_{i=1}^N \text{AP}_i \quad (8)$$

4.1.4. Training Details

The number of epochs for network training was set to 250. During the fine-tuning process using the small dataset of labeled indoor-scene logos, we employed a frozen training strategy that freezes the backbone network. Specifically, we froze the parameters of the MobileNetV2 backbone for the first 50 epochs of training, and global training was carried out for the subsequent 200 epochs. Freezing the backbone network in the first 50 epochs

is aimed at leveraging pre-trained features efficiently while reducing the computational load and stabilizing the training process. And, we unfreeze the backbone network in the following 200 epochs for fine-tuning to adapt its learned features more specifically to the detection task, potentially improving model performance.

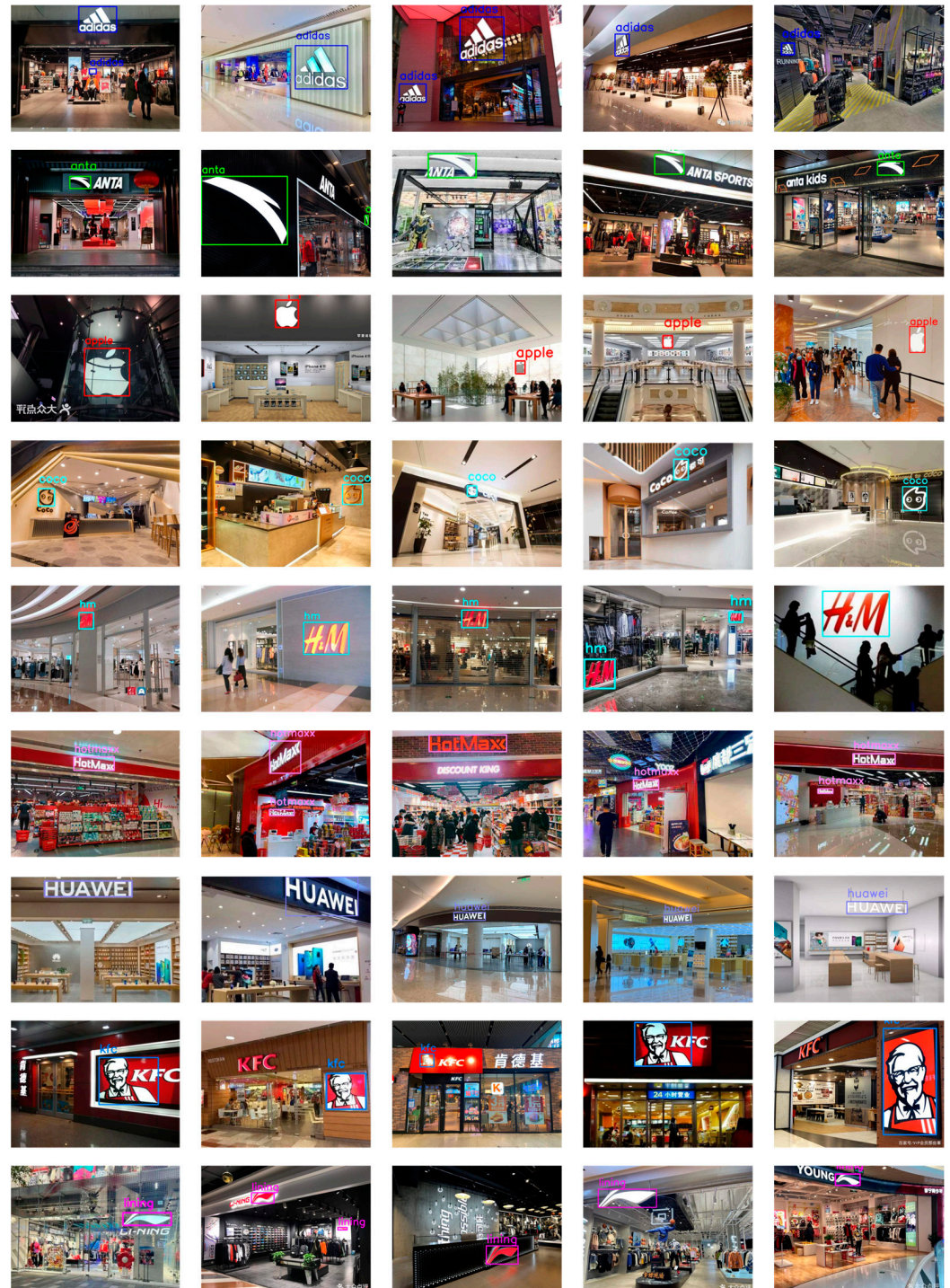


Figure 5. Example images of LOGO2000. First, we created a color dictionary, assigning a different box color to each category. Then, we extracted the coordinates of the top-left and bottom-right corners of the bounding boxes and the label names from each image's corresponding label. We used OpenCV to draw the tilted bounding boxes.

4.1.5. Experimental Protocol

For the MobileNetV2-YOLOv4 network, we conducted two sets of experiments, including loading the UP model from MobileNetV2-AE and random initialization. In addition to the MobileNetV2-YOLOv4 network used in our experiments, we also compared it with commonly used baseline methods in logo object-detection tasks, including Faster R-CNN, YOLOv4, and SSD. Furthermore, to verify the effectiveness of our UP scheme, we conducted ablation experiments on the MobileNetV2-SSD network.

4.2. Results

4.2.1. Learning Curve

We verified the convergence of the model. The comparison of model loss curves between MobileNetV2-YOLOv4 and three baseline methods is shown in Figure 6.

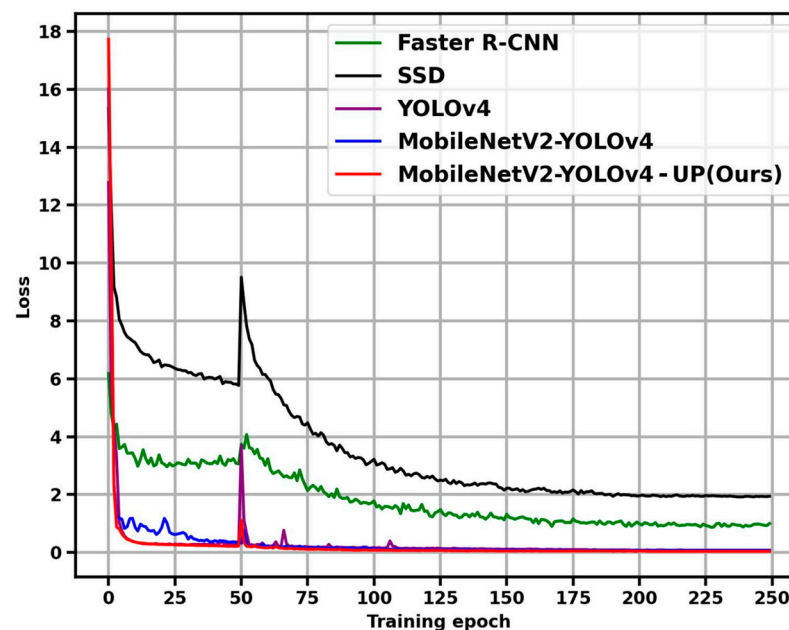


Figure 6. The learning curve of logo-detection networks.

We first analyze the loss curves from the local. Although we used a pre-trained model from MobileNetV2-AE, while other methods did not, the initial loss of our method (MobileNetV2-YOLOv4-UP) is the highest among all models, which may be related to task inconsistency between MobileNetV2-AE (reconstruction) and MobileNetV2-YOLOv4 (detection). For all methods, the loss of the model experienced a sharp decrease in the first 50 epochs, with our method showing the biggest decrease. This proves that the weights from MobileNetV2-AE reduce the difficulty of model regression. However, the loss of the model rebounded after the backbone parameters were unfrozen. We suspect this may be related to the use of frozen training. Our method owned the smallest rebound, indicating that the backbone network had achieved better regression. Nevertheless, the loss of all models quickly decreased again in the following 50 epochs and converged in the subsequent 150 epochs. For our proposed MobileNetV2-YOLOv4-UP, there was still some rebound in the loss of the model due to the differences between UP and the few-shot fine-tuning task in terms of both training optimization objectives and datasets. However, the loss of the model gradually converged in the subsequent training process.

Overall, due to the huge difference in model structure, the loss curves of SSD and Faster-RCNN are quite different from the YOLO series. Therefore, we compared the three methods of the YOLO series, and our method showed a smaller loss, which can also illustrate the effectiveness of UP. From the perspective of model convergence, all four methods achieved good learning effects for logos in large indoor scenes. But from the

perspective of model loss value, our proposed method achieved the best few-shot learning effect, which may be attributed to the good prior information provided by UP conducted on a large unlabeled dataset.

4.2.2. Logo-Detection Performance

The loss curves can not fully and accurately reflect the performance of these models on real-world detection tasks; hence, we need to further evaluate the detection performance of these methods on logos in real indoor scenes. Table 1 presents the mAP scores of three baseline methods (YOLOv4, Faster R-CNN, and SSD), the non-pretrained MobileNetV2-YOLOv4 method, and our proposed MobileNetV2-YOLOv4-UP, on different logo categories of LOGO2000. We conducted the evaluation on a test dataset from LOGO2000 consisting of 600 images that were not used in training.

Table 1. Logo-detection results on the LOGO2000 dataset.

Classes	YOLOv4	Faster-RCNN	SSD	MV2-YOLOv4	MV2-YOLOv4-UP
UNIQLO	72.6	87.2	77.1	69.9	85.3
H&M	97.0	96.1	99.8	95.9	100
ZARA	64.6	84.7	83.8	92.2	87.2
NAIXUE	24.6	42.3	69.5	27.2	52.5
CoCo	61.2	83.2	81.8	62.6	93.2
KFC	60.6	76.1	62.3	50.1	79.5
McDonald's	83.7	88.6	88.0	85.1	87.4
Starbucks	82.2	90.8	84.6	83.9	91.2
Apple	56.1	68.1	68.0	63.7	76.1
OPPO	60.8	82.1	89.3	80.1	95.5
vivo	88.7	84.9	97.5	88.1	94.1
HUAWEI	24.7	52.4	61.5	27.2	70.5
Xiaomi	86.7	100.0	87.0	95.0	97.9
adidas	50.8	58.6	48.3	47.4	72.5
Nike	29.4	58.6	73.5	56.7	71.7
LI-NING	54.2	77.8	89.3	70.3	96.7
ANTA	41.5	64.3	69.8	47.0	62.1
PUMA	56.9	76.3	74.2	76.4	77.0
HotMaxx	45.3	60.6	88.2	55.5	85.2
Watsons	76.4	97.2	96.0	94.1	100
Average	60.9	76.5	79.5	68.4	83.8

¹ The best performance is shown in bold; the second-best performance is shown in blue; ² MV2 is short for MobileNetV2; ³ Columns 2 and 5 in the table constitute an ablation study to validate the effectiveness of using MobileNetV2 as the backbone for YOLOv4.

All five methods in Table 1 were trained on LOGO2000 in a supervised way, with the backbone network of MobileNetV2-YOLOv4-UP initialized with pre-trained weights, while the other methods were trained from scratch. The pre-training of the backbone network in MobileNetV2-YOLOv4-UP was achieved through unsupervised learning on LOGO625 using an autoencoder. Therefore, only MobileNetV2-YOLOv4-UP utilized unsupervised learning.

The experimental results demonstrate that our method is superior to the traditional methods. According to Table 1, our method achieved the best detection performance (bold) in 11 out of 20 categories and the second-best performance (blue) in 7 categories. Although we only achieved the third-best performance in the “McDonald's” and “ANTA” categories, the difference between our results and the best performance was only 1.2% and 7.7% in terms of mAP, respectively. Therefore, in most categories, our method performs quite well. Overall, our method achieved the best detection performance, with a mAP of 83.8%, which is a 4.3% improvement over the best performance (SSD, mAP: 79.5%) among the other four methods. Compared to YOLOv4 with a mAP of 60.9%, which only differs in the backbone

network, our network achieved a 22.9% improvement in mAP. For MobileNetV2-YOLOv4, which has the same backbone network and model structure as our method, our UP strategy achieved a 15.4% improvement.

The effectiveness of backbone and UP has also been confirmed. We first analyzed the results in Columns 2 and 5 in Table 1. We have observed that MobileNetV2-YOLOv4 achieved a 7.5% improvement over the general YOLOv4 model in mAP performance, which indicates that MobileNetV2 is better suited for logo feature extraction compared to CSPDarknet53. Another observation is that, in all 20 categories, our method showed improvement in 19 categories compared to MobileNetV2-YOLOv4, and some categories showed significant improvements, such as “HUAWEI”, which improved from 27.2% to 70.5%, and “HotMaxx”, which improved from 55.5% to 85.2%, demonstrating that our UP can provide useful prior information for the network.

Furthermore, we also obtained some interesting findings. We found that all of the five methods used in the experiment performed at a high level for certain categories, such as “H&M”, possibly due to the more distinctive features and unique structures (a combination of artistic fonts and graphics) of these logos, which can reduce the interference of the background on detection. We also found that detection performance is independent of the shape category (letter-based or graph-based) of the logo. Although letter-based logos lack rich features from human visual perception compared to graph-based logos, they can also achieve good detection performance in the experiment. However, we found that logos with simple geometric structures, such as rectangles (e.g., NAIXUE), perform poorly. This is because the network struggles to distinguish them from other geometric objects in indoor scenes. We also analyzed the impact of LOGO2000’s sample distribution on the detection performance of different logo categories. Since the LOGO2000 dataset contains an equal number of images for each category, similar scenes in the images, and a close number of logos, we believe that the sample distribution in the dataset is unlikely to be the main cause of this performance difference.

In most object-detection tasks, a common approach is to fine-tune the backbone network after pre-training. Typically, MobileNetV2 is pre-trained on the ImageNet dataset. Therefore, we compared MobileNetV2 with UP (ours) to MobileNetV2 pre-trained on ImageNet, as shown in Table 2, which indicates that the mAP performance of the two methods is similar. Notably, our method is designed for scenarios where there is a significant lack of labeled data, making it difficult to obtain large labeled datasets like ImageNet for backbone pre-training. Hence, our method is highly meaningful in such scenarios.

Table 2. Comparison between ImageNet pre-training and our UP.

Methods	mAP
MobileNetV2-YOLOv4 (ImageNet pre-training)	84.1
MobileNetV2-YOLOv4-UP (Ours)	83.8

In the experiment, the backbone network of the control group was supervised pre-trained on ImageNet, while our method was unsupervised pre-trained.

4.2.3. Logo-Detection Efficiency

Table 3 presents the detection speed of the five methods on the cloud server platform. Compared to the two-stage Faster-RCNN, the rest of the four one-stage methods achieved much faster detection speed due to their simpler model structure and data-processing flow. Among them, our proposed method showed a relatively small difference in detection speed compared to YOLOv4 and MV2-YOLOv4. The introduction of the lightweight backbone MobileNetV2 has brought a 7.48% improvement over the general YOLOv4, with a single image-detection time of only 0.0248 s and an FPS score of 40.370. This indicates that our method exhibits promising potential for application in real-time perception on robot platforms with computing devices or cloud service.

Table 3. Evaluation of logo-detection speed.

Methods	FPS	Time
YOLOv4	37.559	0.0266
Faster R-CNN	15.211	0.0657
SSD	98.640	0.0101
MobileNetV2-YOLOv4	41.481	0.0241
MobileNetV2-YOLOv4-UP (Ours)	40.370	0.0248

Only MobileNetV2-YOLOv4-UP employed unsupervised learning during the pre-training of its backbone network.

4.3. Ablation Study

To further demonstrate the effectiveness of our UP strategy, we designed the ablation experiments. Our experiments in 4.2 had already included part of the ablation experiment. To show that the improvement brought by UP is not a random result, we also conducted similar experiments on the MobileNetV2-SSD network by replacing the backbone network of SSD with MobileNetV2. The purpose of doing so is to maintain the consistency of UP for loading the same pre-training weights from MobileNetV2-AE and to investigate the impact of pre-training weights on detection performance. Although there are big differences in network structure between SSD and YOLOv4, to maintain encoding consistency, the backbone network of MobileNetV2-SSD outputs the same three levels of features (bottleneck3, bottleneck5, bottleneck7) as MobileNetV2-YOLOv4.

Table 4 shows the results of the ablation experiments based on UP for both the MobileNetV2-SSD and MobileNetV2-YOLOv4 networks. The significant improvement in detection performance brought by UP is evident for both MobileNetV2-SSD and MobileNetV2-YOLOv4 networks. For MobileNetV2-SSD, the detection performance was improved from 72.1% to 82.0% in terms of mAP, while for MobileNetV2-YOLOv4, it increased from 68.4% to 83.8%. Therefore, what we obtained from UP is a universal logo representation, rather than a local fitting result. Moreover, the performance improvement of MobileNetV2-YOLOv4 (15.4%) was greater than that of MobileNetV2-SSD (9.9%), and the detection performance of MobileNetV2-YOLOv4 (83.8%) was also higher than that of MobileNetV2-SSD (82.0%). These results not only demonstrate the effectiveness of our UP strategy but also suggest that MobileNetV2-YOLOv4 is better suited for extracting and post-processing logo features and exhibiting a more comprehensive absorption of prior knowledge compared to MobileNetV2-SSD networks.

Table 4. Ablation study on UP.

Methods	mAP
MobileNetV2-SSD	72.1
MobileNetV2-SSD-UP	82.0
MobileNetV2-YOLOv4	68.4
MobileNetV2-YOLOv4-UP (Ours)	83.8

Only MobileNetV2-YOLOv4-UP and MobileNetV2-SSD-UP employed unsupervised learning during the pre-training of its backbone network.

5. Conclusions

In this paper, we propose a novel logo-detection method named MobileNetV2-YOLOv4-UP, which integrates unsupervised learning with few-shot learning to address the challenge of logo detection in indoor scenes with limited labeled data. Our approach applies unsupervised learning and few-shot learning to an autoencoder and an object-detection network, respectively. Specifically, we design an autoencoder to capture latent feature representations of logos and perform UP on a large unlabeled dataset. By combining a lightweight backbone network with a one-stage object-detection network, we incorporate the prior information obtained from the autoencoder. This method requires updating the network only on a small dataset of labeled indoor-scene logos, without repetition of UP when new

logos appear, thereby significantly reducing the training complexity. The experimental results of both a publicly available dataset and our self-collected dataset demonstrate the superiority of our approach compared to classical object-detection methods. Additionally, our UP strategy proves to be effective, leading to significant performance improvements. Future work may focus on enhancing the autoencoder and refining the training strategy of unsupervised learning to further improve the performance of this method on small datasets.

Author Contributions: Conceptualisation, S.Z.; Formal analysis, C.Y. and Q.Y.; Supervision, S.Z. and Q.Y.; Writing—original draft, C.Y. and Q.Y.; Writing—review and editing, C.Y., Q.Y. and Z.Y. All authors will be informed about each step of manuscript processing including submission and revision. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Natural Science Foundation of Shanghai (No. 22ZR1465700).

Data Availability Statement: The datasets used in this experiment can be accessed at the following address: logo625: <https://pan.baidu.com/s/12xmAXEibxdODunGAghsaeA>. (permanent validity). password: 73hy. LOGO2000: https://drive.google.com/file/d/1wg0PLIrY2HdDm2QhfR_OmuHLdfgPIMup/view?usp=drive_link. (permanent validity).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zeng, Z.; Lin, H.; Kang, Z.; Xie, X.; Yang, J.; Li, C.; Zhu, L. A Semantics-Guided Visual Simultaneous Localization and Mapping with U-Net for Complex Dynamic Indoor Environments. *Remote Sens.* **2023**, *15*, 5479. [\[CrossRef\]](#)
2. Liao, Z.; Hu, Y.; Zhang, J.; Qi, X.; Zhang, X.; Wang, W. So-slam: Semantic object slam with scale proportional and symmetrical texture constraints. *IEEE Robot. Autom. Lett.* **2022**, *7*, 4008–4015. [\[CrossRef\]](#)
3. Lin, S.; Wang, J.; Xu, M.; Zhao, H.; Chen, Z. Contour-SLAM: A robust object-level SLAM based on contour alignment. *IEEE Trans. Instrum. Meas.* **2023**, *72*, 1–12. [\[CrossRef\]](#)
4. Sadeghi Esfahlani, S.; Sanaei, A.; Ghorabian, M.; Shirvani, H. The deep convolutional neural network role in the autonomous navigation of mobile robots (SROBO). *Remote Sens.* **2022**, *14*, 3324. [\[CrossRef\]](#)
5. Schulz, R.; Talbot, B.; Lam, O.; Dayoub, F.; Corke, P.; Upcroft, B.; Wyeth, G. Robot navigation using human cues: A robot navigation system for symbolic goal-directed exploration. In Proceedings of the 2015 IEEE International Conference on Robotics and Automation (ICRA), Seattle, WA, USA, 26–30 May 2015; pp. 1100–1105.
6. Ye, C.; Qian, X. 3-D object recognition of a robotic navigation aid for the visually impaired. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2017**, *26*, 441–450. [\[CrossRef\]](#)
7. Hinton, G.E.; Salakhutdinov, R.R. Reducing the dimensionality of data with neural networks. *Science* **2006**, *313*, 504–507. [\[CrossRef\]](#)
8. Boia, R.; Bandrabur, A.; Florea, C. Local description using multi-scale complete rank transform for improved logo recognition. In Proceedings of the 2014 10th International Conference on Communications (COMM), Bucharest, Romania, 29–31 May 2014; pp. 1–4.
9. Boia, R.; Florea, C.; Florea, L. Elliptical asift agglomeration in class prototype for logo detection. In Proceedings of the BMVC, Swansea, UK, 7–10 September 2015; pp. 115.111–115.112.
10. Bagdanov, A.D.; Ballan, L.; Bertini, M.; Del Bimbo, A. Trademark matching and retrieval in sports video databases. In Proceedings of the International Workshop on Workshop on Multimedia Information Retrieval, Augsburg, Germany, 24–29 September 2007; pp. 79–86.
11. Zhang, X.; Zhang, D.; Liu, F.; Zhang, Y.; Liu, Y.; Li, J. Spatial HOG based TV logo detection. In Proceedings of the Fifth International Conference on Internet Multimedia Computing and Service, Huangshan, China, 17–18 August 2013; pp. 76–81.
12. Li, K.-W.; Chen, S.-Y.; Su, S.; Duh, D.-J.; Zhang, H.; Li, S. Logo detection with extendibility and discrimination. *Multimed. Tools Appl.* **2014**, *72*, 1285–1310. [\[CrossRef\]](#)
13. Hoi, S.C.; Wu, X.; Liu, H.; Wu, Y.; Wang, H.; Xue, H.; Wu, Q. Logo-net: Large-scale deep logo detection and brand recognition with deep region-based convolutional networks. *arXiv* **2015**, arXiv:1511.02462.
14. Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [\[CrossRef\]](#)
15. Li, Y.; Shi, Q.; Deng, J.; Su, F. Graphic logo detection with deep region-based convolutional networks. In Proceedings of the 2017 IEEE Visual Communications and Image Processing (VCIP), St. Petersburg, FL, USA, 10–13 December 2017; pp. 1–4.
16. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1440–1448.
17. Chen, H.; Li, X.; Wang, Z.; Hu, X. Robust logo detection in e-commerce images by data augmentation. In Proceedings of the 29th ACM International Conference on Multimedia, Virtual, 20–24 October 2021; pp. 4789–4793.

18. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards real-time object detection with region proposal networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1137–1149. [\[CrossRef\]](#)
19. Yin, K.; Hou, S.; Li, Y.; Li, C.; Yin, G. A real-time vehicle logo detection method based on improved YOLOv2. In Proceedings of the Wireless Algorithms, Systems, and Applications: 15th International Conference, WASA 2020, Qingdao, China, 13–15 September 2020; pp. 666–677.
20. Redmon, J.; Farhadi, A. YOLO9000: Better, faster, stronger. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July; 2017; pp. 7263–7271.
21. Yang, S.; Zhang, J.; Bo, C.; Wang, M.; Chen, L. Fast vehicle logo detection in complex scenes. *Opt. Laser Technol.* **2019**, *110*, 196–201. [\[CrossRef\]](#)
22. Redmon, J.; Farhadi, A. Yolov3: An incremental improvement. *arXiv* **2018**, arXiv:1804.02767.
23. Wang, J.; Min, W.; Hou, S.; Ma, S.; Zheng, Y.; Jiang, S. Logodet-3k: A large-scale image dataset for logo detection. *ACM Trans. Multimed. Comput. Commun. Appl. (TOMM)* **2022**, *18*, 1–19. [\[CrossRef\]](#)
24. Jiang, X.; Sun, K.; Ma, L.; Qu, Z.; Ren, C. Vehicle Logo Detection Method Based on Improved YOLOv4. *Electronics* **2022**, *11*, 3400. [\[CrossRef\]](#)
25. Bochkovskiy, A.; Wang, C.-Y.; Liao, H.-Y.M. Yolov4: Optimal speed and accuracy of object detection. *arXiv* **2020**, arXiv:2004.10934.
26. Orti, O.; Tous, R.; Gomez, M.; Poveda, J.; Cruz, L.; Wust, O. Real-time logo detection in brand-related social media images. In Proceedings of the Advances in Computational Intelligence: 15th International Work-Conference on Artificial Neural Networks, IWANN 2019, Gran Canaria, Spain, 12–14 June 2019; pp. 125–136.
27. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.-Y.; Berg, A.C. Ssd: Single shot multibox detector. In Proceedings of the Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, 11–14 October 2016; pp. 21–37.
28. Zhang, J.; Chen, L.; Bo, C.; Yang, S. Multi-scale vehicle logo detector. *Mob. Netw. Appl.* **2021**, *26*, 67–76. [\[CrossRef\]](#)
29. Hou, Q.; Min, W.; Wang, J.; Hou, S.; Zheng, Y.; Jiang, S. Foodlogodet-1500: A dataset for large-scale food logo detection via multi-scale feature decoupling network. In Proceedings of the 29th ACM International Conference on Multimedia, Virtual, 20–24 October 2021; pp. 4670–4679.
30. Su, H.; Gong, S.; Zhu, X. Multi-perspective cross-class domain adaptation for open logo detection. *Comput. Vis. Image Underst.* **2021**, *204*, 103156. [\[CrossRef\]](#)
31. Li, X.; Hou, S.; Zhang, B.; Wang, J.; Jia, W.; Zheng, Y. Long-range dependence involutional network for Logo Detection. *Entropy* **2023**, *25*, 174. [\[CrossRef\]](#)
32. Jain, R.K.; Watasue, T.; Nakagawa, T.; Takahiro, S.; Iwamoto, Y.; Xiang, R.; Yen-Wei, C. LogoNet: Layer-aggregated attention centernet for logo detection. In Proceedings of the 2021 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 10–12 January 2021; pp. 1–6.
33. Yang, Z.; Liao, H.; Zhang, H.; Li, W.; Xia, J. Representation based few-shot learning for brand-logo detection. In Proceedings of the 2022 IEEE 4th International Conference on Power, Intelligent Computing and Systems (ICPICS), Shenyang, China, 29–31 July 2022; pp. 350–354.
34. Su, H.; Zhu, X.; Gong, S. Open logo detection challenge. *arXiv* **2018**, arXiv:1807.01964.
35. Huh, M.; Agrawal, P.; Efros, A.A. What makes ImageNet good for transfer learning? *arXiv* **2016**, arXiv:1608.08614.
36. Chen, H.; Wang, Y.; Wang, G.; Qiao, Y. Lstd: A low-shot transfer detector for object detection. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
37. Mandic, D.P. A generalized normalized gradient descent algorithm. *IEEE Signal Process. Lett.* **2004**, *11*, 115–118. [\[CrossRef\]](#)
38. Fei, H.; Li, Q.; Sun, D. A survey of recent research on optimization models and algorithms for operations management from the process view. *Sci. Program.* **2017**, *2017*, 7219656. [\[CrossRef\]](#)
39. Yan, L.; Zheng, Y.; Cao, J. Few-shot learning for short text classification. *Multimed. Tools Appl.* **2018**, *77*, 29799–29810. [\[CrossRef\]](#)
40. Santoro, A.; Bartunov, S.; Botvinick, M.; Wierstra, D.; Lillicrap, T. Meta-learning with memory-augmented neural networks. In Proceedings of the International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 1842–1850.
41. Lopez-Martin, M.; Carro, B.; Sanchez-Esguevillas, A. Variational data generative model for intrusion detection. *Knowl. Inf. Syst.* **2019**, *60*, 569–590. [\[CrossRef\]](#)
42. Li, X.; Liang, Y.; Zhao, M.; Wang, C.; Jiang, Y. Few-Shot Learning with Generative Adversarial Networks Based on WOA13 Data. *Comput. Mater. Contin.* **2019**, *60*, 1073. [\[CrossRef\]](#)
43. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016; pp. 770–778.
44. Pfister, T.; Charles, J.; Zisserman, A. Domain-adaptive discriminative one-shot learning of gestures. In Proceedings of the Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, 6–12 September 2014; pp. 814–829.
45. Sinaga, K.P.; Yang, M.-S. Unsupervised K-means clustering algorithm. *IEEE Access* **2020**, *8*, 80716–80727. [\[CrossRef\]](#)
46. Zhang, T.; Ramakrishnan, R.; Livny, M. BIRCH: An efficient data clustering method for very large databases. *ACM Sigmod Rec.* **1996**, *25*, 103–114. [\[CrossRef\]](#)
47. Ester, M.; Kriegl, H.-P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. *KDD* **1996**, *96*, 226–231.
48. Elgohary, A.; Farahat, A.K.; Kamel, M.S.; Karray, F. Embed and conquer: Scalable embeddings for kernel k-means on mapreduce. In Proceedings of the 2014 SIAM International Conference on Data Mining, Philadelphia, PA, USA, 24–26 April 2014; pp. 425–433.

49. Schubert, E.; Sander, J.; Ester, M.; Kriegel, H.P.; Xu, X. DBSCAN revisited, revisited: Why and how you should (still) use DBSCAN. *ACM Trans. Database Syst. (TODS)* **2017**, *42*, 1–21. [\[CrossRef\]](#)
50. Tian, L.; Fan, C.; Ming, Y.; Jin, Y. Stacked PCA network (SPCANet): An effective deep learning for face recognition. In Proceedings of the 2015 IEEE International Conference on Digital Signal Processing (DSP), Singapore, 21–24 July 2015; pp. 1039–1043.
51. Kalman, D. A singularly valuable decomposition: The SVD of a matrix. *Coll. Math. J.* **1996**, *27*, 2–23. [\[CrossRef\]](#)
52. Comon, P. Independent component analysis, a new concept? *Signal Process.* **1994**, *36*, 287–314. [\[CrossRef\]](#)
53. Hinton, G. Stochastic neighbor embedding. *Adv. Neural Inf. Process. Syst.* **2003**, *15*, 857–864.
54. Rumelhart, D.E.; Hinton, G.E.; Williams, R.J. Learning Internal Representations by Error Propagation, Parallel Distributed Processing, Explorations in the Microstructure of Cognition, ed. DE Rumelhart and J. McClelland. Vol. 1. 1986. *Biometrika* **1986**, *71*, 599–607.
55. LeCun, Y.; Touresky, D.; Hinton, G.; Sejnowski, T. A theoretical framework for back-propagation. In *Proceedings of the 1988 Connectionist Models Summer School*; M. Kaufmann: San Mateo, CA, USA, 1988; pp. 21–28.
56. Liou, C.-Y.; Cheng, W.-C.; Liou, J.-W.; Liou, D.-R. Autoencoder for words. *Neurocomputing* **2014**, *139*, 84–96. [\[CrossRef\]](#)
57. Smolensky, P. Information Processing in Dynamical Systems: Foundations of Harmony Theory. 1986. Available online: https://www.researchgate.net/publication/239571798_Information_processing_in_dynamical_systems_Foundations_of_harmony_theory (accessed on 1 June 2023).
58. Geng, Z.; Li, Z.; Han, Y. A new deep belief network based on RBM with glial chains. *Inf. Sci.* **2018**, *463*, 294–306. [\[CrossRef\]](#)
59. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, *27*, 2672–2680.
60. Chen, T.; Kornblith, S.; Norouzi, M.; Hinton, G. A simple framework for contrastive learning of visual representations. In Proceedings of the International Conference on Machine Learning, Virtual, 13–18 July 2020; pp. 1597–1607.
61. He, K.; Fan, H.; Wu, Y.; Xie, S.; Girshick, R. Momentum contrast for unsupervised visual representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 13–19 June 2020; pp. 9729–9738.
62. Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In Proceedings of the 2009 IEEE Conference on Computer Vision and Pattern Recognition, Miami, FL, USA, 20–25 June 2009; pp. 248–255.
63. Bar, A.; Wang, X.; Kantorov, V.; Reed, C.J.; Herzig, R.; Chechik, G.; Rohrbach, A.; Darrell, T.; Globerson, A. Detreg: Unsupervised pretraining with region priors for object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, New Orleans, LA, USA, 18–24 June 2022; pp. 14605–14615.
64. Wang, X.; Zhang, R.; Shen, C.; Kong, T.; Li, L. Dense contrastive learning for self-supervised visual pre-training. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3024–3033.
65. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. Mobilenetv2: Inverted residuals and linear bottlenecks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
66. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In Proceedings of the Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, 5–9 October 2015; pp. 234–241.
67. Michelucci, U. An introduction to autoencoders. *arXiv* **2022**, arXiv:2201.03898.
68. Dwivedi, K.; Roig, G. Representation similarity analysis for efficient task taxonomy & transfer learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12387–12396.
69. Liu, S.; Kong, W.; Chen, X.; Xu, M.; Yasir, M.; Zhao, L.; Li, J. Multi-scale ship detection algorithm based on a lightweight neural network for spaceborne SAR images. *Remote Sens.* **2022**, *14*, 1149. [\[CrossRef\]](#)

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.