electronics

MDPI

Article

Automatic Modulation Classification with Neural Networks via Knowledge Distillation

Shuai Wang and Chunwu Liu *

College of Intelligent Science, National University of Defense Technology, Changsha 410073, China; ws17626045941@163.com

* Correspondence: liuchunwu@nudt.edu.cn

Abstract: Deep learning is used for automatic modulation recognition in neural networks, and because of the need for high classification accuracy, deeper and deeper networks are used. However, these are computationally very expensive for neural network training and inference, so its utility in the case of a mobile with memory limitations or weak computational power is questionable. As a result, a trade-off between network depth and network classification accuracy must be considered. To address this issue, we used a knowledge distillation method in this study to improve the classification accuracy of a small network model. First, we trained Inception–Resnet as a teacher network, which has a size of 311.77 MB and a final peak classification accuracy of 93.09%. We used the method to train convolutional neural network 3 (CNN3) and increase its peak classification accuracy from 79.81 to 89.36%, with a network size of 0.37 MB. It was also used similarly to train mini Inception–Resnet and increase its peak accuracy from 84.18 to 93.59%, with a network size of 39.69 MB. When we compared all classification accuracy peaks, we discover that knowledge distillation improved small networks and that the student network had the potential to outperform the teacher network. Using knowledge distillation, a small network model can achieve the classification accuracy of a large network model. In practice, choosing the appropriate student network based on the constraints of the usage conditions while using knowledge distillation (KD) would be a way to meet practical needs.

Keywords: modulation recognition; knowledge distillation; teacher-student framework; convolutional neural network

check for updates

Citation: Wang, S.; Liu, C. Automatic Modulation Classification with Neural Networks via Knowledge Distillation. *Electronics* **2022**, *11*, 3018. https://doi.org/10.3390/electronics11193018

Academic Editor: Sung Jin Yoo

Received: 27 August 2022
Accepted: 18 September 2022
Published: 22 September 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2022 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

A wealth of specialized expertise has become available in the field of communications due to the rapid development of communication technology, which ensures the accurate transfer of data. It is a complex and mature engineering field with many distinct areas of investigation that have all seen diminishing returns in improved performance, particualrly on the physical layer [1]. Modulation types of communication signals diversify and become more complex along with the wireless communication environment, which puts increased demand on the modulation identification of signals. Deep learning is brought into signal modulation identification by employing convolutional neural network (CNN) approach to identify the modulation types of signals in order to further explore and solve the problem of modulation recognition.

New classification problems have arisen as a result of emerging wireless technology, which means that automatic modulation classification (AMC) in real-world environments continues to be a dynamic research field [2]. A DL-based (or processing block) that does not require a mathematically tractable model and channel might be able to optimize the function of a communications system better [1], so deep learning is useful for solving the problem of new modulation recognition. Initially, neural networks in combination with complex-valued temporal data, was investigated, and it was demonstrated that network depth does not limit wireless modulation recognition. Therefore, and it was suggested that future research focus on synchronization and equalization. Refs. [3–5] Deep learning in

underwater communication scenarios, as well as multipath scenarios with blind channels for collaborative modulation classification problems, have been investigated. Refs. [6,7] The researchers continued to use deep networks, introduced advanced networks, changed the input data, and used spectrograms to improve recognition accuracy, achieving a peak classification accuracy of 93% [8–11].

The Refs. [12] mainly focused on the mixed data of the primary modulation and remodulation signals and studied the modulation recognition algorithm based on a combination of a CNN and a signal cyclic spectrum image. An edge intelligence algorithm of a CNN, based on an attention mechanism, was developed to carry out modulation recognition (MR) of the edge signal and bring MR closer to the antenna terminal [13]. Snoap et al. considered the problem of deep-learning-based classification of digitally modulated signals using I/Q data and studied the generalization ability of a trained neural network (NN) to classify digitally modulated signals correctly [14].

Deeper and deeper networks are used to pursue high classification MR accuracy, along with large computations and model storage. In practical inference, a balance between classification accuracy and network model size should be maintained. The model should be as small as possible, so the effect can be as good as possible. Current AMC research is based primarily on very deep networks, and while these models achieve high performance accuracy, the high cost of training these models raises concerns about their suitability for mobile deployment.

With the development of the NPU (Neural network Processing Unit), massively parallel processing architectures with distributed memory architectures—such as graphic processing units (GPUs) and increasingly specialized chips for NN inference [15]—have proven to be energy efficient and capable of impressive computational throughput when fully used by concurrent algorithms [16]. Engineers and developers frequently have to work within a limited time budget in industrial and commercial scenarios [17], so the NN model size, computing effort, and operating time must all be taken into account. The goal is to classify automatic modulation accurately in the simplest way possible using accessible and less expensive devices such as smartphones or embedded devices.

Knowledge distillation(KD) is used in this paper to improve the accuracy of small models to allow for faster and smaller memory occupation during the inference deployment phase. In KD, a high-precision large-scale network is first trained, and then it is used to induce the training of the small network to improve its classification accuracy, which is also known as "compression of the large model" or "knowledge migration" [18].

In various machine learning applications—object detection, acoustical modeling and natural language processing—KD has been used successfully [19]. Some studies are related to our work with KD in real-life scenarios (e.g., skin cancer classification [20], autonomous vehicles [21], drowsiness detection [22] and human activity recognition) [23]. The applications of KD are hardly explored in the AMC domain, and how KD translates in it remains an open question. If KD could be used efficiently, it would reduce the complexity and computing expense considerably. There are also other model complexity reduction methods such as quantization [24] and pruning [25]. Quantization approximates a trained neural network with low bit width numbers to reduce the memory requirement and computing cost. It is also applicable after training deep NNs via pruning or KD. In pruning, an NN is made smaller and efficient by eliminating the values of the weight tensors to derive a computationally cost-efficient model. It is an expensive process since it requires multiple pruning iterations to get the final model [26].

In this paper, we first investigated modulation recognition using CNNs then designed Inception–Resnet deformation networks that can achieve high accuracy classification. The classification accuracy of the student networks improved by using KD.To pick the most appropriate values for the hyperparameters, we used repeated adjustments of the hyperparameters T and α. When the best outcomes were compared to the network without KD, the network with it had higher classification accuracy while maintaining the same model size.

The classification accuracy of NNs with three convolutional layers improved with high-precision networks through knowledge distillation. Work related to neural networks and knowledge distillation is introduced in Section 2. Experiments are set up in Section 3 to train and test the proposed scheme, and the results were compared to select the best result. In Section 4, the results of modulation recognition classification accuracy are compared when KD was used and when it was not.

2. Materials and Methods

2.1. Basic Principle of Signal Modulation

Modulation identification means that for a given receive signal $r_{(t)}, 0 <= t <= T$, from the set consisting of C possible modulation types $\{\omega_1, \omega_2, \cdots, \omega_c\}$, the modulation type of $r_{(t)}$ is selected and identified. AMC combines signal processing and pattern recognition, but because communication signals and channel noise are typically modeled as stochastic processes, they are coupled with unknown signal fading, multipath propagation, and interference effects. As a result, modulation mode identification is essentially another multiple unknown parameter with AMC, and it is very important in both collaborative and non-collaborative domains.

If the channel is assumed to be ideal for a general signal model and carrier and timing synchronization are not taken into account, the modulated signal can be uniformly modeled as

$$r^{(i)}(t) = [I^i(t) + jQ^i(t)]e^{j(2\pi f_c t + \theta)} + n_{(t)}, 0 <= t <= T, \tag{1}$$

where i is the modulation type denote; $r^{(i)}(t)$ is the received complex signal; t denotes the simulation time; $n(t)$ denotes the noise during signal transmission; $I^{(i)}(t)$, $Q^{(i)}(t)$ denotes the in-phase and quadrature components of the low-pass equivalent signal, respectively; f_c is the carrier frequency; θ is the initial phase of the carrier; and T is the observed signal duration.

2.2. SNR and accuracy

In digital signal processing (DSP) we deal with extremely big numbers and extremely small numbers together (e.g., the strength of a signal compared to the strength of the noise). The logarithmic scale of a dB lets us have more dynamic range when we express numbers or plot them.

For a given value x, we can represent x in dB using the following formula:

$$x_{db} = 10log_{10}x. \tag{2}$$

The signal-to-noise ratio (SNR) is what we use measure the differences in strength between the signal and noise, and in practice it is almost always in dB, in practice.

$$SNR = P_{signal}/P_{noise} \tag{3}$$

$$SNR_{dB} = P_{signal\_dB} - P_{noise\_dB} \tag{4}$$

If someone says "SNR = 0 dB" it means the signal and noise power are the same. A positive SNR means our signal is stronger than the noise, while a negative SNR means that the noise is stronger. Detecting signals at a negative SNR is usually pretty tough.

To calculate accuracy, one needs to find the true positive (TP), true negative (TN), false positive (FP) and false negative (FN) values. True positive indicates that the output of the real class is yes, and the output of the predicted class is also yes, whereas true negative indicates that the value of the real class and the value of the anticipated class are no. False positive indicates that the real class is no while the predicted class is yes, whereas a false negative indicates that the real class is yes but the expected class is no.

Because all samples were retrieved in this paper, only TP and FP were required: TP indicated that the modulation predicted to be this modulation was this modulation. FP

indicated that what was predicted to be a modulation was actually another. The accuracy metrics is the ratio of correct predictions over the total number of predictions evaluated

$$Accuracy = TP/TP + FP \tag{5}$$

2.3. CNN

A CNN is focused on handling data that has a grid-like structure. It shows that at least one layer of the network uses special linear operations called convolutions rather than the more common matrix multiplications [27].

In CNN terminology, the first parameter x of the convolution is usually called the input and the second parameter w is called the kernel function. The structure of the output is called the feature mapping. In general, when a computer processes data, time is discretized so that the moment t can only take integer values, and assuming that both x and w are defined at the integer moment t [27]. The convolution in discrete form is defined as Equation (6):

$$s(t) = (x * \omega)(t) = \sum_{a=-\infty}^{\infty} x(a)\omega(t-a). \tag{6}$$

In machine learning,the kernel is usually a multidimensional array of parameters that have been optimized by a learning algorithm, and input data are mostly arrays. Generally, these multidimensional arrays are referred to as tensors. Convolutional operations are generally carried out in multiple dimensions in practical uses. For example, a two-dimensional image I is taken as input and convolution is performed using a two-dimensional kernel K as in Equation (7).

$$S(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(m,n)K(i-m, j-n). \tag{7}$$

The convolution is exchangeable, and we can equivalently write

$$S(i,j) = (K * I)(i,j) = \sum_{m}\sum_{n} I(i-m, j-n)K(m,n). \tag{8}$$

Convolutional operations can be used interchangeably because the relative inputs of the kernels can be flipped. The libraries of neural networks, however, typically use intercorrelation functions, which are nearly identical to convolutional operations but don't flip the kernels as (9), in the application of neural networks:

$$S(i,j) = (I * K)(i,j) = \sum_{m}\sum_{n} I(i+m, j+n)K(m,n), \tag{9}$$

Figure 1 depicts a two-dimensional convolution. The convolution kernel is a 2×2 matrix, and the input data is a 3×4 matrix. A matrix of the same size as the convolution kernel moves over the input data, which is multiplied by the data at the corresponding position of the convolution kernel, and the product is added to produce a result. The convolution result of the input data and the convolution kernel is represented by the result matrix. Convolution refers to an operation that consists of multiple parallel kernals, because a convolution with a single kernel can only extract one type of feature, despite the fact that the kernel acts on multiple spatial locations, and we usually want each layer of the network to extract multiple types of features at multiple locations [27].

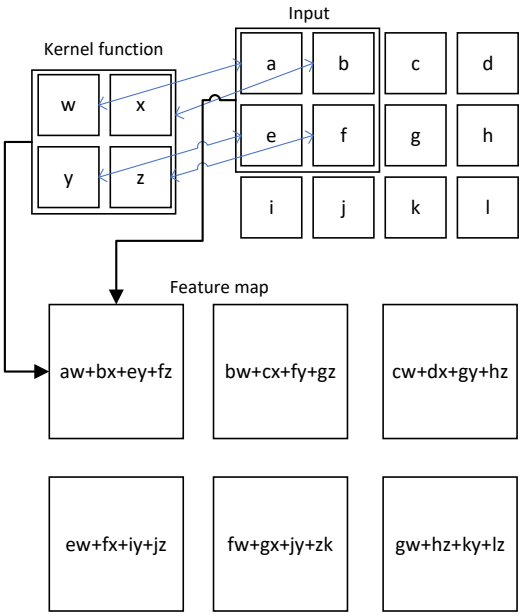


Figure 1. Two-dimensional convolution without kernel flipping.

A CNN, which is a deep-learning algorithm, can automatically classify and identify features without any human intervention [28]. A visible trend in NNs for classification is building deeper networks to learn more complex functions and hierarchical feature relationships. Deep networks enable more complex functions to be learned more readily from raw data [29]. Deep neural networks are typically used in three steps to solve modulation classification problems. The first step is to design the network architecture. The next is to train the network to select weights that minimize loss. The third is to validate and test the network to solve the problem [30].

2.4. Residual Network

When deeper networks are able to start converging, a degradation problem is exposed. When the network depth increases, accuracy becomes saturated (which might be unsurprising) and then degrades rapidly. Unexpectedly, such degradation is not caused by overfitting, and adding more layers to a suitably deep model leads to higher training errors. The Refs. [31] addressed the degradation problem by introducing a deep residual learning framework. Instead of hoping that each few stacked layers directly fit a desired underlying mapping, they explicitly let these layers fit a residual mapping.

A deep residual learning framework is depicted in Figure 2. Following the activation function, the input data matrix was processed in two parallel ways: one, to leave the data unchanged and the other, to feed the data through two convolutional layers, and finally the two parallel ways are summed and fed into the activation function. The original function becomes Equation (10):

$$y = \mathcal{F}(x, \{W_i\}) + x, \tag{10}$$

where x and y are the input and output vectors of the layers. The function $\mathcal{F}(x, \{W_i\})$ represents the residual mapping to be learned. The dimensions of x and $\mathcal{F}$ must be equal. If this is not the case, we can perform a linear projection Ws by shortcut connections to match the dimensions as Equation (11):

$$y = \mathcal{F}(x, \{W_i\}) + W_s x, \tag{11}$$

Ws is only used when matching dimensions [31].

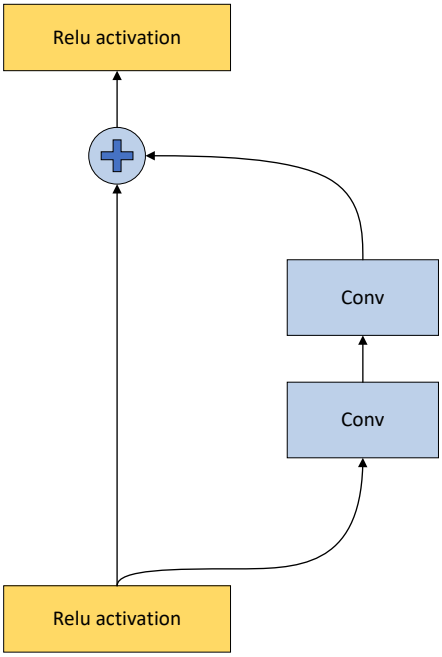


Figure 2. Residual network diagram.

Residual connections are inherently necessary for training very deep convolutional models becasue they seem to improve the training speed greatly, which is a great argument for their use [32].

2.5. Inception

The main idea of the Inception architecture is based on finding out how an optimal local sparse structure in a convolutional vision network can be approximated and covered by readily available dense components [33].

The Inception module with dimension reductions is depicted in Figure 3. It receives data from the previous layer and processes it in four parallel ways into different convolution channels for calculation; if the convolution channel is smaller than the input channel, the data is downscaled. Finally, all output data channels are stitched together to produce the final result. 1×1 convolutions are used to compute reductions before the expensive 3×3 and 5×5 convolutions. Besides being used as reductions, they also include the use of rectified linear activation which makes 1×1 convolutions dual purpose: most critically, they were used mainly as dimension reduction modules to remove computational bottlenecks that would otherwise have limited the size of our networks. This allowed not just an increase in depth, but also in the width of our networks without a significant performance penalty [33].

The generous use of dimensional reduction and parallel structures of the Inception modules mitigated the effect of structural changes on nearby components [34].

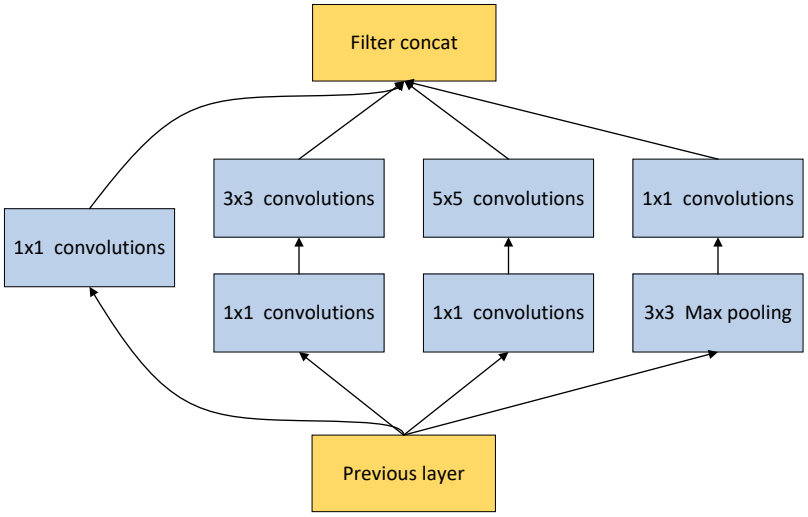


Figure 3. Inception module with dimension reductions.

2.6. Knowledge Distillation

The main idea behind knowledge distillation is that to achieve superior performance, the student model must imitate the teacher model. Knowledge types, distillation techniques and teacher–student learning architecture all play critical roles in student learning. [19]. A general teacher–student framework for knowledge distillation is shown in Figure 4. The teacher-student framework is the most fundamental framework for knowledge distillation. The teacher network in this paper is the pre-trained Inception–ResNet, and the student networks are a CNN with three convolutional layers and a Mini-Inception–ResNet. The output of the teacher network is distilled to form a soft and hard target to calculate the student network's loss function.

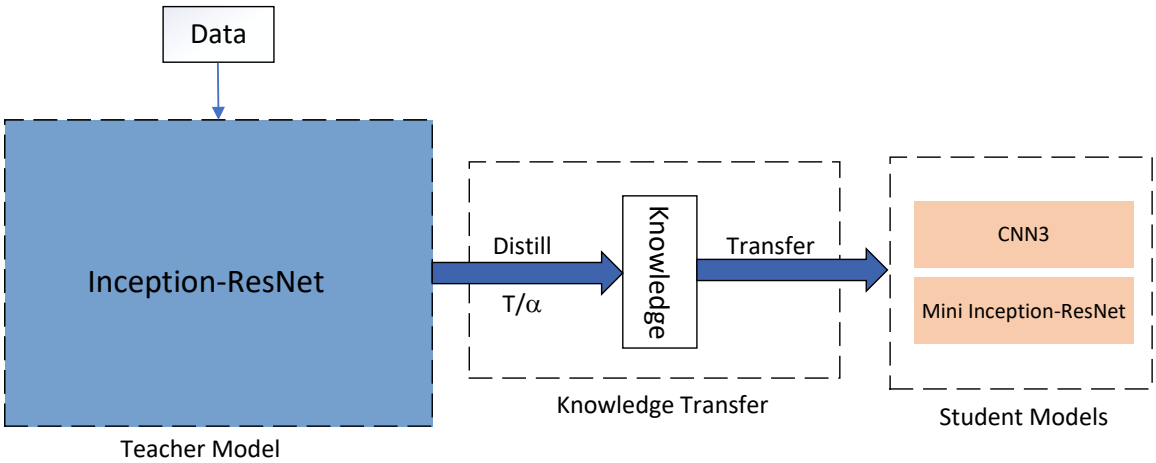


Figure 4. The generic teacher–student framework for knowledge distillation [19].

Neural networks typically produce class probabilities by using a "softmax" output layer that converts the logit z_i, computed for each class into a probability q_i, by comparing z_i with the other logits [18].

$$q_i = \frac{exp(z_i/T)}{\sum_j exp(z_j/T),} \tag{12}$$

where T is a temperature that is normally set to 1. Using a higher value for T produces a softer probability distribution over the classes. To understand the knowledge distillation, a benchmark model, which is the distillation combined with student losses, is given in Figure 5. The basic process of knowledge extraction is as follows: The same input enters the

teacher and student networks; the temperature of the softmax layer of the teacher network is set to t; the output produced is soft targets; the temperature of the softmax layer of the student network is set to t; and the soft targets produced together with those produced by the teacher network produce a distillation loss; the temperature of the student network's softmax layer is set to 1; and the resulting soft targets and true labels of the data generate the both the student and distillation losses as well as a combined loss, which comprises the overall loss.

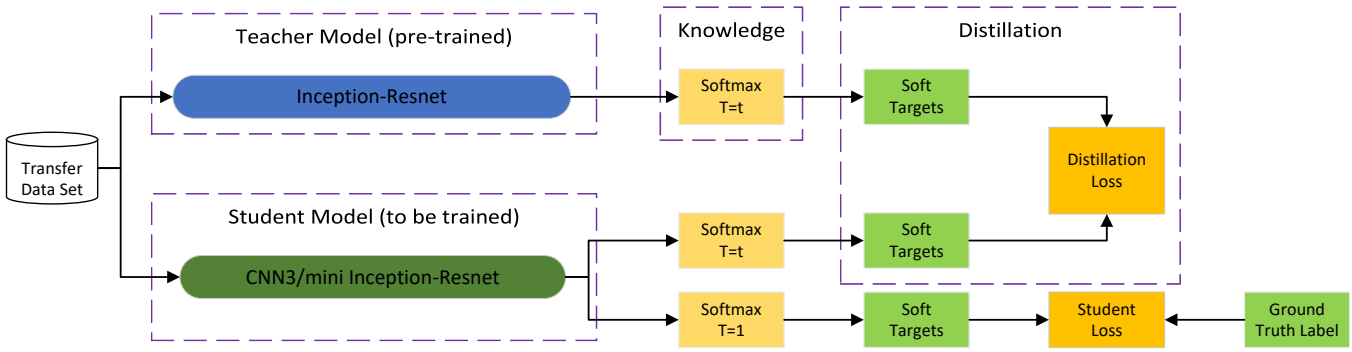


Figure 5. The specific architecture of the benchmark knowledge distillation.

When calculating losses in KD, the weighted average of two different objective functions is used. The first is cross entropy with the soft targets, which is computed using the same high temperature in the softmax of the distilled model that was used to generate the soft targets from the cumbersome model. The second is cross entropy with the correct labels. Since the magnitudes of the gradients produced by the soft target scale as $1/T^2$, it is important to multiply them by T^2 when using both hard and soft targets [19].

The process of knowledge distillation is shown in Figure 5, and its loss function [35] is

$$L_{KD}(W_{student}) = \alpha T^2 * CrossEntropy(Q_S^\tau, Q_T^\tau) + (1-\alpha) * CrossEntropy(Q_S, Y_{true},) \quad (13)$$

where T and α are hyperparameters, T refers to the temperature of distillation, and α refers to the proportion of soft loss in the total loss.

3. Experiments

3.1. Structure of the Teacher and Student Network

The structure of the Teacher network shown in Figure 6 is based on Inception–Resnet, which adapted to the size of the dataset used in this paper by varying the size of the convolutional kernels over the number of sizes. Inception–ResnetA was repeated 10 times; Inception–ResnetB, 20 times; and Inception–ResnetC 10 times.

There are two student networks: mini-Inception–Resnet and CNN3. The Figure 7 depicts the network structure of the mini-Inception–Resnet. The input to it was computed in the following order: stem module, Inception–ResnetA, reduction-A, Inception–ResnetB, reduction-B, and Inception–ResnetC, followed by pooling and softmax. The CNN3 used three convolutional networks for classification as shown in Figure 8, and is the simple equivalent of an entry-level network with fast inference, small computation, small number of parameters, and small space occupied by the model.

The input to Inception–Resnet was computed in the following order: stem module, 10 tandem Inception–ResnetA, reduction-A, 10 tandem Inception–ResnetB, reduction-B, and 10 tandem Inception–ResnetC, followed by pooling and softmax. The structures of $Inception--ResnetA$, $Inception--ResnetB$, and $Inception--ResnetC$ used in the paper are shown in the Figure 9, respectively. Inception–Resnet contains the *Stem* network module showed in Figure 10, the three $Inception--Resnet$ modules, and two *Reduction* modules.

All three Inception–Resnet modules have a similar structure. After the previous level's input was passed through the ReLU activation function, it was divided into two parallel paths: one was completely unchanged; the other passed through several parallel convolutional layers, then through a final 1×1 convolution into the same number of channels as the initial input data. Finally the outputs of the two paths were summed and passed through the relu activation function into the next module. Using the $Inception--Resnet$ module has the advantage of being a fast converging network, and from the results the Inception–Resnet network had a good classification effect [32].

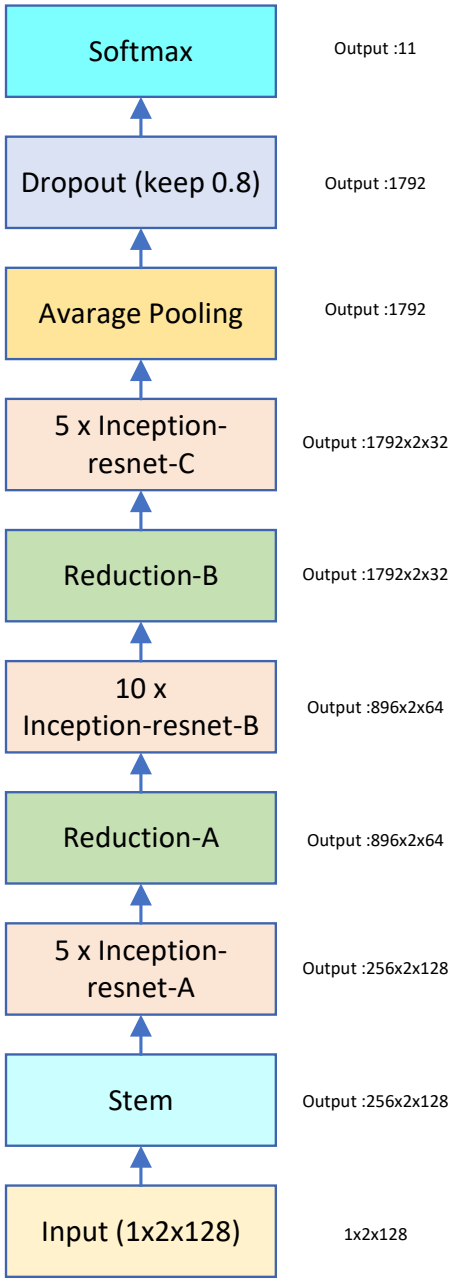


Figure 6. Inception–Resnet.

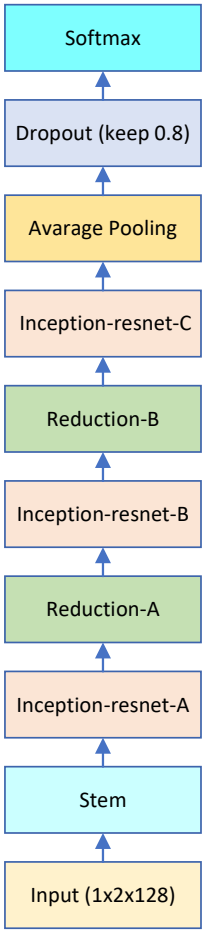


Figure 7. mini Inception–Resnet.

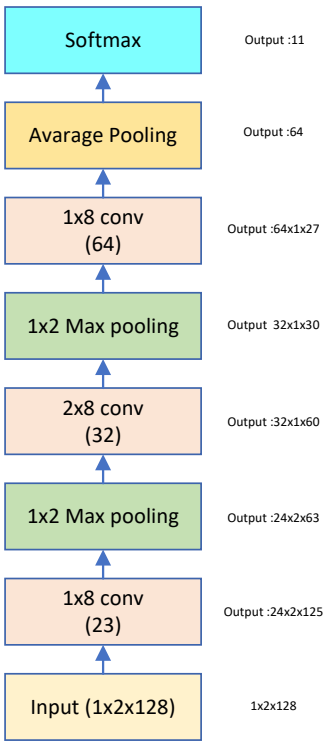
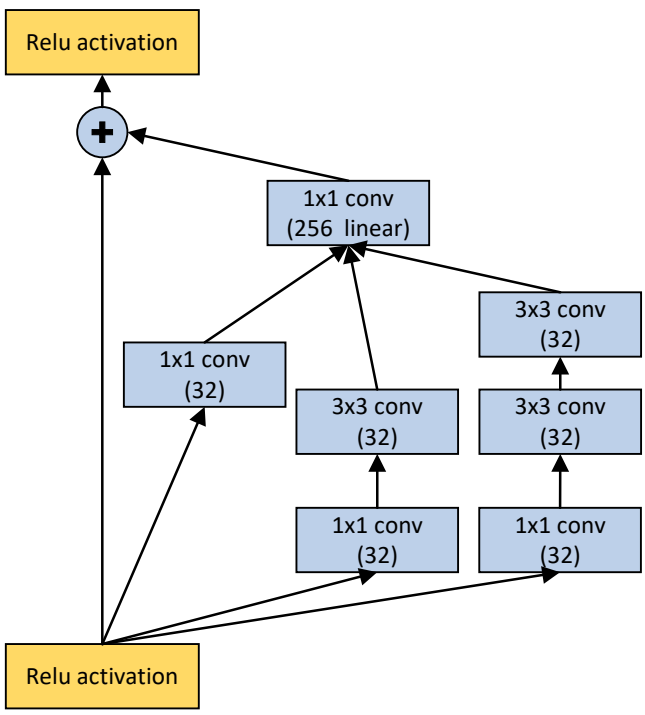
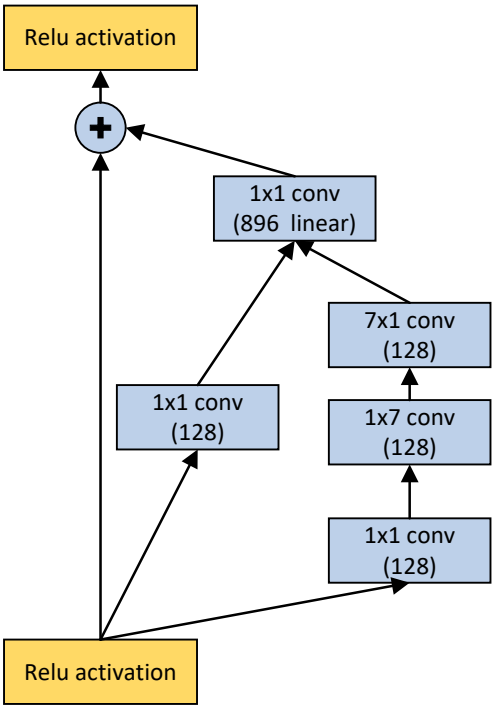


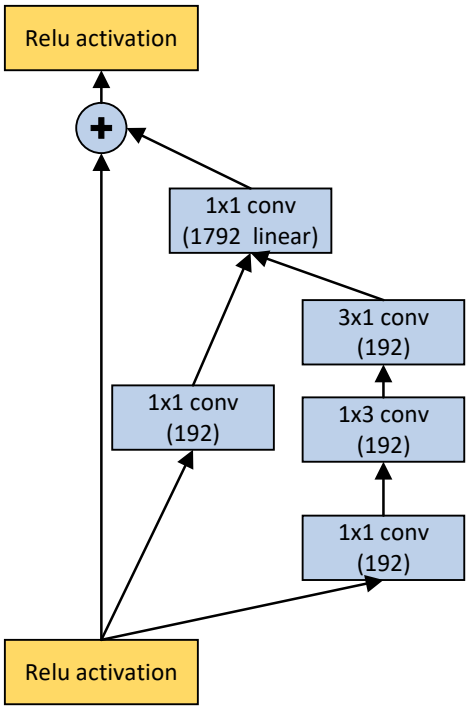
Figure 8. CNN3.



(**a**)Inception–ResnetA



(**b**)Incepion–ResnetB



(**c**)Inception–ResnetC

Figure 9. The schema for the three Inception–Resnet modules of the teacher network.

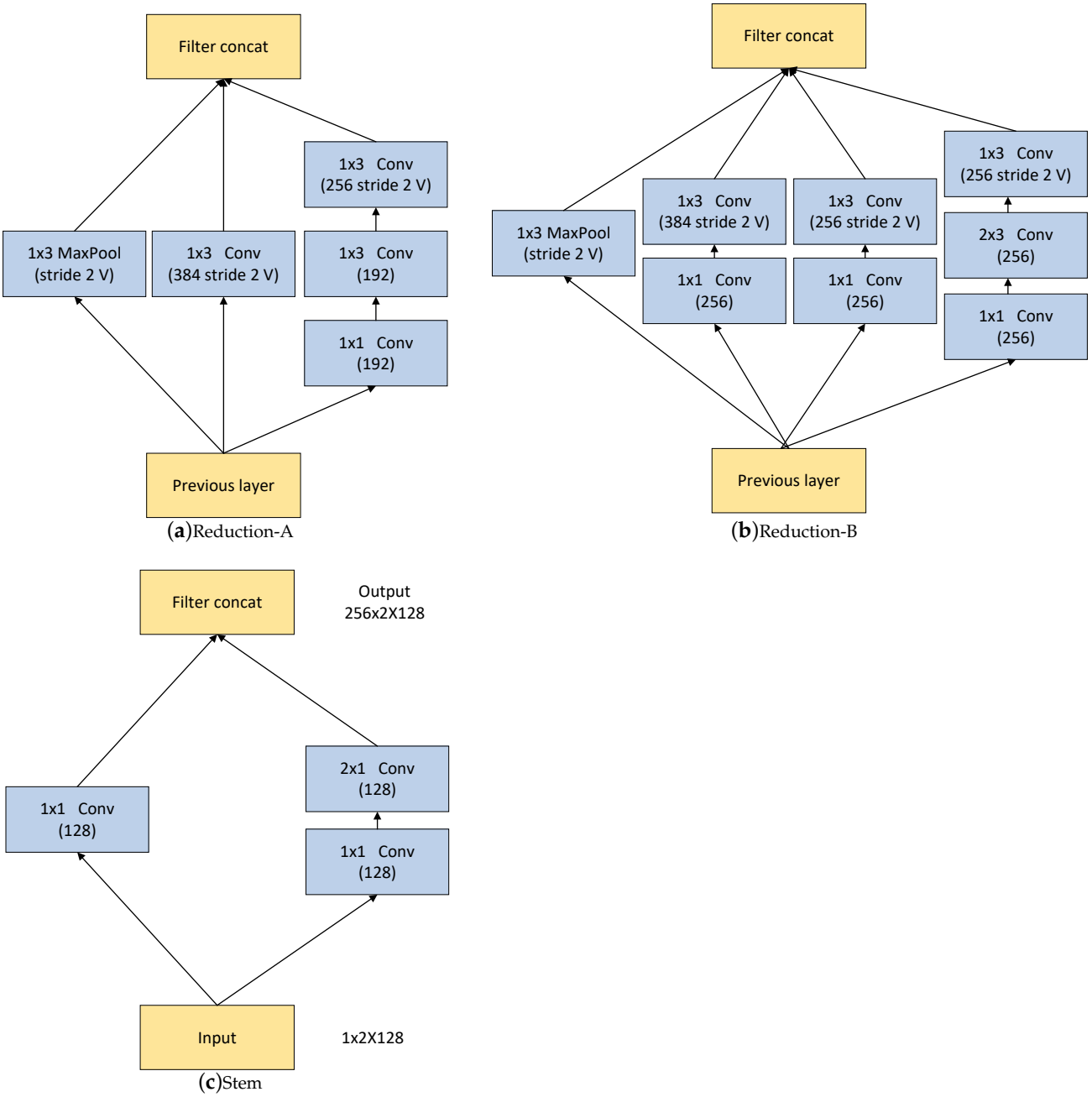


Figure 10. Structures of reduction blocks and Stem.

The article pre-trained the teacher network and the undistilled student network. It used the Adam optimizer in training and the NatchNorm and ReLU activation functions after all convolutional layers. The loss function is the categorical cross-entropy function, and it is seen from the Figure 11 that the classification accuracy of the teacher network is higher than 90% when the signal-to-noise ratio is higher than 0 dB, while it can be seen that the accuracy of the undistilled student network had about 80% classification accuracy at a high SNR, which was much less than the teacher network becasue its structure is much smaller than that of the teacher network.

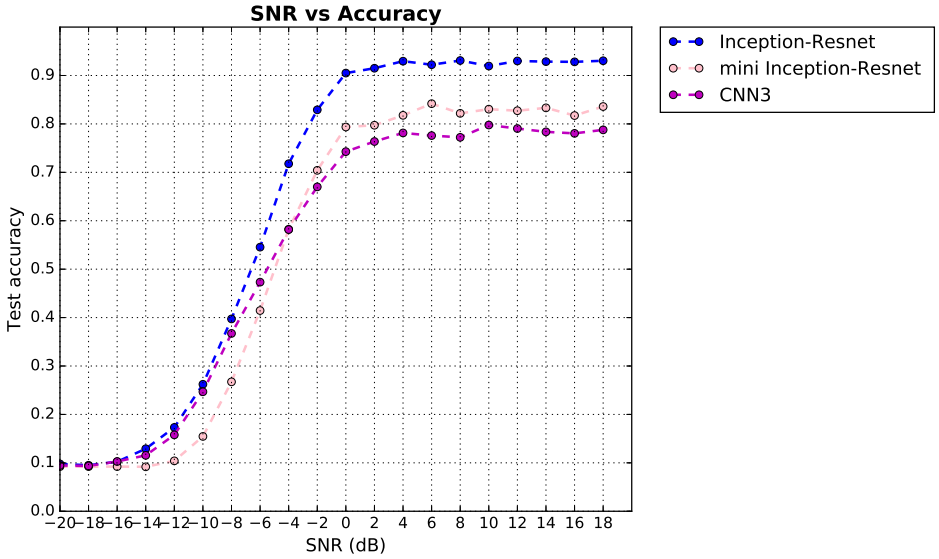


Figure 11. The classification accuracy of teacher and student network.

3.2. Loss Function

The loss function Equation (13) designed for knowledge distillation was used in the experiment. The teacher network was based on the Inception–Resnet network, and the student network was a three-layer convolutional network. The soft target was the output of Inception–Resnet, which corresponded to Q_T, and the output of the three-layer convolutional network corresponded to Q_s.

3.3. Dataset and Training

This paper used the RadioML 2016.10b dataset [2] as the basis for evaluating the modulation recognition task. The dataset consisted of 11 modulations: 8 digital and 3 analog, all of which are widely used in wireless communications systems all around us. These consist of BPSK, QPSK, 8PSK, 16QAM, 64QAM, BFSK, CPFSK, and PAM4 for digital; and WB-FM, AM-SSB, and AM-DSB for analog modulations. Details about the generation of this dataset can be found in [36] . Data was modulated at a rate of roughly 8 samples per symbol with a normalized average transmit power of 0 dB [3]. The dataset was split into two sections for the experiment, with 20% serving as a validation set and the remaining 80% serving as training data. Adam was the optimizer used in this paper. Its batch size is 512, and learning rate is 0.001.

When training in the experiments, an early stop mechanism was employed to halt training when performance on the validation dataset began to deteriorate. The ability of the deep neural network to generalize was improved by stopping the training before the neural network overfitted the training dataset. The network was initially configured with a minimum loss of 100. When the network's loss on the validation set was less than the minimum loss, the network structure was saved at this point, and the loss at this time was noted as the minimum loss. Training was stopped when the network's loss on the validation set exceeded the minimum loss for 10 consecutive iterations.

3.4. Experimental Procedure

The general framework of the experimental process in this paper is shown in Figure 12, where the teacher network is the high precision network trained in advance and the student network is the network to be trained. The experimental procedure is the same as that of knowledge distillation: first, the soft loss is calculated using the knowledge distillation process, then the hard loss is calculated, and the two are added to yield the total loss.

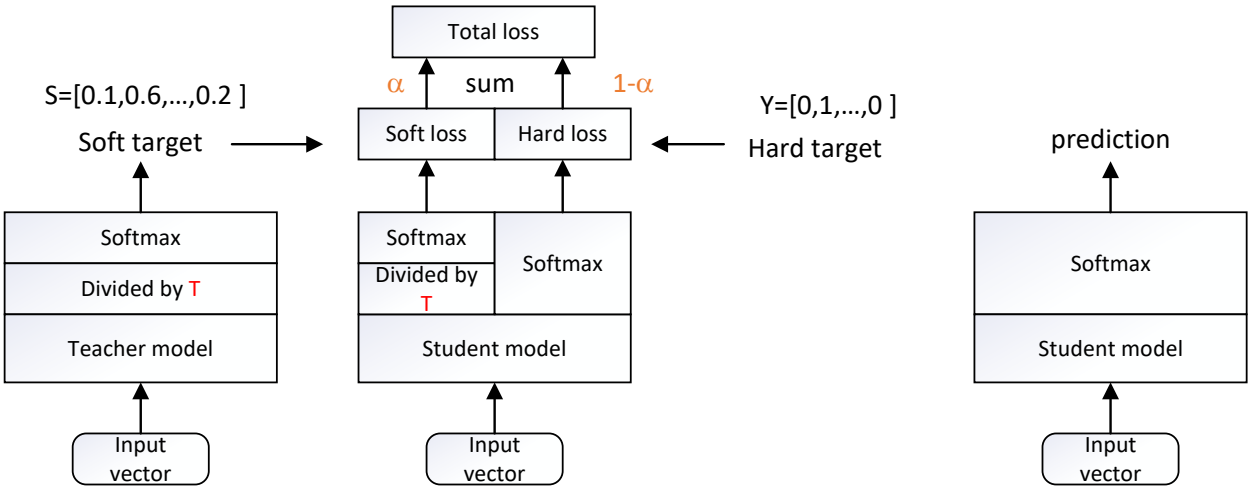


Figure 12. The process of knowledge distillation.

Typically, large amounts of data are essential for training the deep learning model to avoid overfitting. There are many parameters in deep neural networks, so if there is not enough data to training them, they tend to remember the entire training set, which will result in good training, but bad performance on testing set [37].

The weighted average of the soft and hard losses is the knowledge distillation loss. Soft loss was calculated first by inputting the dataset to the teacher and student networks and dividing the result by the temperature parameter T, followed by a softmax calculation to obtain the probability distribution of softening. The output of the teacher and student networks was then used to calculate KL divergence to obtain a soft loss. The hard target was the true marker of the sample, which can be represented by a one-hot vector inputted into the student network to get an output. Then softmax was calculated, and the cross-entropy loss was calculated using the softmax result and the true marker of the sample.

The experiment raised the temperature from 1 to 9, with a temperature difference of 2. The proportion α of the total loss changed from 0.1 to 0.9 in each temperature, and the difference between α was 0.1, so there were nine α in each temperature. There were five temperatures in total, each temperature and α formed a small experiment, and each small experiment got a distillation student network for a total of 45.

4. Result and Analysis

4.1. Evaluation of Classification

Figures 13 and 14 display the KD effects that can be produced by "α" at various temperatures. Assuming that a model's quality is determined by its highest classification accuracy over all SNRs, it is clear that, after knowledge distillation, the DSCNN3 (CNN3 after knowledge distillation) performed best at T = 1, $\alpha = 0.4$, T = 3, $\alpha = 0.7$, T = 5, $\alpha = 0.1$,T = 7, $\alpha = 0.6$,and T = 9, $\alpha = 0.1$, the DSminiIRNET(mini-Inception–Resnet after knowledge distillation) performed best at T = 1, $\alpha = 0.4$, T = 3, $\alpha = 0.7$, T = 5, $\alpha = 0.8$, T = 7, $\alpha = 0.6$, and T = 9, $\alpha = 0.5$.

It can be seen that most networks after KD achieved higher classification accuracy, and the classification accuracy of networks after knowledge distillation were very high at the right temperature and α.

Figures 13f and 14f represent the accuracy of the best model at each temperature. The best DSCNN3 appears at T=1 and $\alpha = 0.4$, which corresponded to the network model. The best DSminiIRNET appeared at T=7 and $\alpha = 0.6$, which also corresponds to the network model.

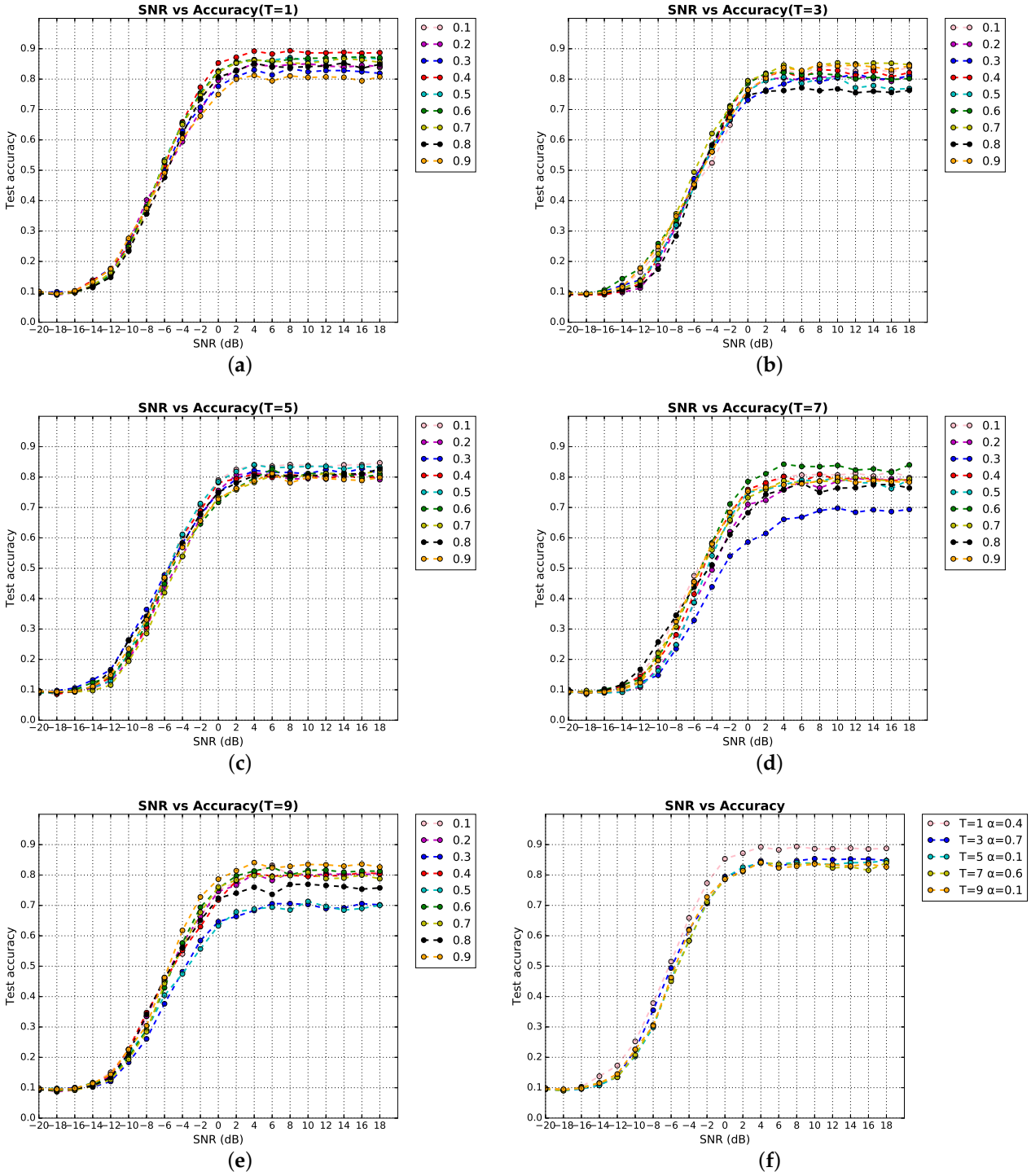


Figure 13. CNN3 classification accuracy after knowledge distillation with different parameters. (**a**) Classification accuracy of student networks corresponding to different α at T = 1; (**b**) at T = 3; (**c**) at T = 5; (**d**) at T = 7; (**e**) at T = 9; and (**f**), the best classification accuracy in each T.

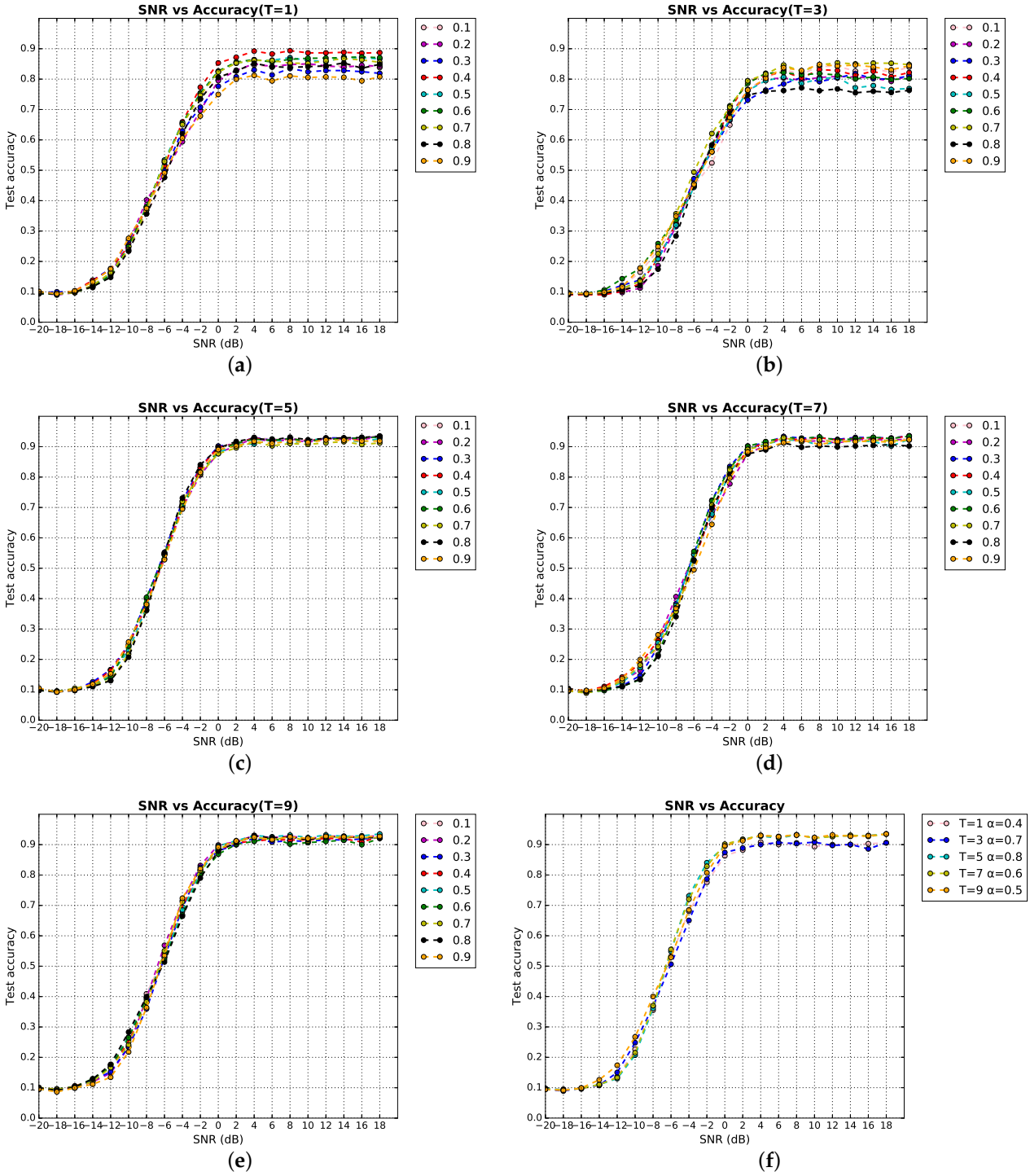


Figure 14. Mini Inception–Resnet classification accuracy after knowledge distillation with different parameters. (**a**) Classification accuracy of student networks corresponding to different α at T = 1; (**b**) at T = 3; (**c**) at T = 5; (**d**) at T = 7; (**e**) at T = 9; (**f**), the best classification accuracy in each T.

Figure 15 depicts the optimal DSCNN3 classification accuracy. Once the SNR exceeded 0 dB, the network's classification accuracy became flat, and the classification accuracy of CNN3 was stable at around 78% and that of DSCNN3 around 89%. According to Table 1, CNN3 had a peak classification accuracy of 0.7981; DSCNN3, 0.8936; Inception–Resnet, 0.9309. Without changing the network's size or computational effort, knowledge distillation improved CNN3's peak classification accuracy by 0.1.

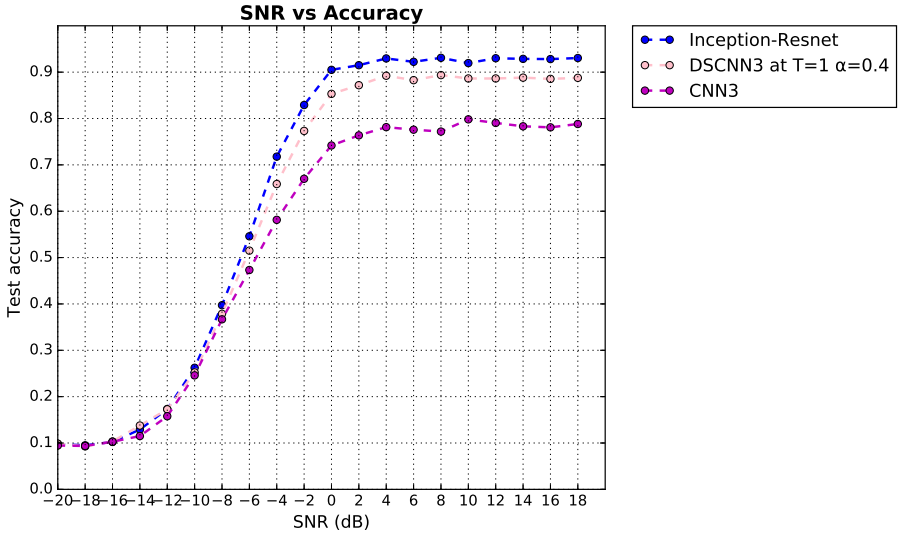


Figure 15. Classification accuracy of teacher network, student network and the best network after knowledge distillation. (DSCNN3 represents CNN3 after knowledge distillation).

Figure 16 depicts the optimal DSCNN3 classification accuracy. Once the SNR exceeded 0 dB, the network's classification accuracy becames flat and the classification accuracy of the mini-Inception-ResNet was stable around 83% and that of DSminiIRNET around 93%. When the SNR was greater than -6 dB, the classification accuracy curves of the DSminiIRNET and the Inception–Resnet overlapped, and the curves of the DSminiIRNET were slightly higher than those of Inception–Resnet. As documented by the Table 1, mini-Inception-ResNet had a peak classification accuracy of 0.8418 and for DSminiIRNET it was 0.9359, which was greater than the classification accuracy peak of the teacher network. Without changing the size of the network or the amount of computation, KD improved the mini-Incepion–ResNet classification accuracy peak by 9.4%.

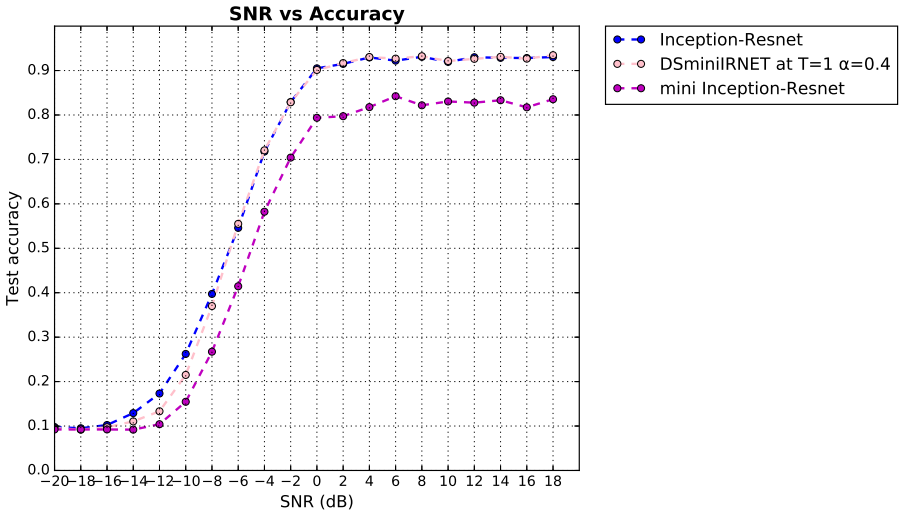


Figure 16. The classification accuracy of the teacher network, student network and the best network after knowledge distillation. (DSminiIRNET represents the mini-Inception–Resnet after knowledge distillation).

Table 1. Teacher network classification accuracy peak and comparison of student network classification accuracy peaks before and after knowledge distillation.

	Inception–Resnet	mini-Inception–Resnet	CNN3
before KD	0.9309	0.8418	0.7981
after KD		0.9359	0.8936

Knowledge distillation can transfer the relationships between different classifications as information to the student network for learning, allowing it to have a higher classification accuracy and outperform the teacher network's smaller models.

4.2. Computation Complexity.

The complexity of an algorithm can be divided into time and space complexity. Time complexity, which is defined as the time of calculation on the algorithm, can be quantitatively analyzed with floating-point operations (FLOPs). Space complexity describes the memory occupation when the algorithm runs [17].

The total model size is the memory occupied by the model. Total parameters indicates the number of them in the different models. FLOPs also indicate the complexity of the deep neural network.

Figure 15 shows that after knowledge distillation, the model's accuracy can be improved without changing its complexity. Table 2 compares the computational complexity of the teacher network to that of the best student network.

Table 2. Computational complexity of the teacher network vs. the student network.

Network	Estimated Total Size (MB)	Total Parameters	FLOPs
Inception–Resnet	311.77	32,353,675	3,232,437,248
mini Inception–Resnet	39.69	3,995,787	449,975,296
CNN3	0.37	30,179	1,257,360

5. Conclusions

In this paper,we proposed a scheme to improve the classification accuracy of small network models for AMC. A highly accurate teacher network induced student network model training to improve accuracy via knowledge distillation.

We conducted experiments to compare the accuracy of student network models obtained by using different hyperparameters T,α in knowledge distillation, from which the best ones were chosen. The peak classification accuracy of the teacher model was 93.09%, and the model size was 311.77 MB; the peak classification accuracy of two student networks after knowledge distillation was 93.59% and 89.36%, and the model sizes were 39.69 and 0.37 MB. Knowledge distillation was successful in reducing model size and improving classification accuracy. When we needed to reduce computational complexity and model size in memory-limited devices or when real-time performance was required, we found that KD was a useful approach for solving this problem.

The use of KD on AMC improved the model's classification accuracy without changing the model size. Comparing model size, parameter size, FLOPS of the student network to those of the teacher network demonstrated the efficacy of knowledge distillation in improving the accuracy of the small network and providing a useful idea for applying AMC in practice. Knowledge distillation can reduce model complexity while improving network classification accuracy. It is useful whether the goal is to pursue high accuracy or reduce complexity. We aim to explore pruning and quantization methods for more details about AMC to further reduce model complexity and training time.

Author Contributions: Conceptualization, S.W.; methodology, S.W.; software, S.W.; validation, S.W.; formal analysis, S.W.; investigation, S.W., C.L.; resources, S.W.; data curation, S.W., C.L.; writing—

original draft preparation, S.W.; writing—review and editing, S.W.; visualization, S.W.; supervision, S.W.; project administration, S.W.; funding acquisition, S.W. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data used in this study are freely available from: https://www.deepsig.ai/datasets (accessed on 26 August 2022).

Conflicts of Interest: The authors declare that they have no known competing financial interest or personal relationships that could have appeared to influence the work reported in this paper.

Abbreviations

The following abbreviations are used in this manuscript:

AMC	Automatic modulation classification
GPUs	Graphic processing units
KD	Knowledge Distillation
DSCNN3	CNN3 after knowledge distillation
DSminiIRNET	mini Inception–Resnet after knowledge distillation

References

1. O'Shea, T.; Hoydis, J. An Introduction to Deep Learning for the Physical Layer. *IEEE Trans. Cogn. Commun. Netw.* **2017**, *3*, 563–575. https://doi.org/10.1109/TCCN.2017.2758370.
2. Dobre, O.; Abdi, A.; Bar-Ness, Y.; Su, W. Survey of automatic modulation classification techniques: Classical approaches and new trends. *IET Commun.* **2007**, *1*, 137. https://doi.org/10.1049/iet-com:20050176.
3. O'Shea, T.J.; Corgan, J.; Clancy, T.C. Convolutional Radio Modulation Recognition Networks. *arXiv* **2016**, arXiv:1602.04105.
4. Kim, B.; Kim, J.; Chae, H.; Yoon, D.; Choi, J.W. Deep neural network-based automatic modulation classification technique. In Proceedings of the 2016 International Conference on Information and Communication Technology Convergence (ICTC), Jeju, Korea, 19–21 October 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 579–582. https://doi.org/10.1109/ICTC.2016.7763537.
5. West, N.E.; O'Shea, T. Deep architectures for modulation recognition. In Proceedings of the 2017 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN), Baltimore, MD, USA, 6–9 March 2017; IEEE: Piscataway, NJ, USA, 2017; pp. 1–6. https://doi.org/10.1109/DySPAN.2017.7920754.
6. Wang, Y.; Zhang, H.; Sang, Z.; Xu, L.; Cao, C.; Gulliver, T.A. Modulation Classification of Underwater Communication with Deep Learning Network. *Comput. Intell. Neurosci.* **2019**, *2019*, 8039632. https://doi.org/10.1155/2019/8039632.
7. Zhang, J.; Cabric, D.; Wang, F.; Zhong, Z. Cooperative Modulation Classification for Multipath Fading Channels via Expectation-Maximization. *IEEE Trans. Wirel. Commun.* **2017**, *16*, 6698–6711. https://doi.org/10.1109/TWC.2017.2728530.
8. Liu, X.; Yang, D.; Gamal, A.E. Deep Neural Network Architectures for Modulation Classification. *arXiv* **2018**, arXiv:1712.00443.
9. O'Shea, T.J.; Roy, T.; Clancy, T.C. Over-the-Air Deep Learning Based Radio Signal Classification. *IEEE J. Sel. Top. Signal Process.* **2018**, *12*, 168–179. https://doi.org/10.1109/JSTSP.2018.2797022.
10. Zeng, Y.; Zhang, M.; Han, F.; Gong, Y.; Zhang, J. Spectrum Analysis and Convolutional Neural Network for Automatic Modulation Recognition. *IEEE Wirel. Commun. Lett.* **2019**, *8*, 929–932. https://doi.org/10.1109/LWC.2019.2900247.
11. Wu, P.; Sun, B.; Su, S.; Wei, J.; Zhao, J.; Wen, X. Automatic Modulation Classification Based on Deep Learning for Software-Defined Radio. *Math. Probl. Eng.* **2020**, *2020*, 2678310. https://doi.org/10.1155/2020/2678310.
12. Liu, R.; Guo, Y.; Zhu, S. Modulation Recognition Method of Complex Modulation Signal Based on Convolution Neural Network. In Proceedings of the 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 11–13 December 2020; IEEE: Piscataway, NJ, USA, 2020; pp. 1179–1184. https://doi.org/10.1109/ITAIC49862.2020.9338875.
13. Jiao, J.; Sun, X.; Zhang, Y.; Liu, L.; Shao, J.; Lyu, J.; Fang, L. Modulation Recognition of Radio Signals Based on Edge Computing and Convolutional Neural Network. *J. Commun. Inf. Netw.* **2021**, *6*, 280–300. https://doi.org/10.23919/JCIN.2021.9549123.
14. Snoap, J.A.; Popescu, D.C.; Spooner, C.M. On Deep Learning Classification of Digitally Modulated Signals Using Raw I/Q Data. In Proceedings of the 2022 IEEE 19th Annual Consumer Communications & Networking Conference (CCNC), Las Vegas, NV, USA, 8–11 January 2022; IEEE: Piscataway, NJ, USA, 2022; pp. 441–444. https://doi.org/10.1109/CCNC49033.2022.9700688.
15. Chen, Y.H.; Krishna, T.; Emer, J.S.; Sze, V. Eyeriss: An Energy-Efficient Reconfigurable Accelerator for Deep Convolutional Neural Networks. *IEEE J. Solid-State Circuits* **2017**, *52*, 127–138. https://doi.org/10.1109/JSSC.2016.2616357.

16. Raina, R.; Madhavan, A.; Ng, A.Y. Large-scale deep unsupervised learning using graphics processors. In Proceedings of the Proceedings of the 26th Annual International Conference on Machine Learning-ICML'09, Montreal, QC, Canada, 14–18 June 2009; ACM Press: Montreal, QC, Canada, 2009; pp. 1–8. https://doi.org/10.1145/1553374.1553486.
17. He, K.; Sun, J. Convolutional Neural Networks at Constrained Time Cost. *arXiv* **2014**, arXiv:1412.1710.
18. Hinton, G.; Vinyals, O.; Dean, J. Distilling the Knowledge in a Neural Network. *arXiv* **2015**, arXiv:1503.02531.
19. Gou, J.; Yu, B.; Maybank, S.J.; Tao, D. Knowledge Distillation: A Survey. *Int. J. Comput. Vis.* **2021**, *129*, 1789–1819. https://doi.org/10.1007/s11263-021-01453-z.
20. Esteva, A.; Kuprel, B.; Novoa, R.A.; Ko, J.; Swetter, S.M.; Blau, H.M.; Thrun, S. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* **2017**, *542*, 115–118.
21. Janveja, I.; Nambi, A.; Bannur, S.; Gupta, S.; Padmanabhan, V. Insight: Monitoring the state of the driver in low-light using smartphones. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* **2020**, *4*, 1–29.
22. Chen, R.; Ai, H.; Shang, C.; Chen, L.; Zhuang, Z. Learning lightweight pedestrian detector with hierarchical knowledge distillation. In Proceedings of the 2019 IEEE International Conference on Image Processing (ICIP), Taipei, Taiwan, 22–25 September 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 1645–1649.
23. Plötz, T.; Guan, Y. Deep learning for human activity recognition in mobile computing. *Computer* **2018**, *51*, 50–59.
24. Nagel, M.; Fournarakis, M.; Amjad, R.A.; Bondarenko, Y.; van Baalen, M.; Blankevoort, T. A white paper on neural network quantization. *arXiv* **2021**, arXiv:2106.08295.
25. Wu, R.T.; Singla, A.; Jahanshahi, M.R.; Bertino, E.; Ko, B.J.; Verma, D. Pruning deep convolutional neural networks for efficient edge computing in condition assessment of infrastructures. *Comput.-Aided Civ. Infrastruct. Eng.***2019**, *34*, 774–789.
26. Khan, M.S.; Alam, K.N.; Dhruba, A.R.; Zunair, H.; Mohammed, N. Knowledge distillation approach towards melanoma detection. *Comput. Biol. Med.* **2022**, *146*, 105581.
27. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
28. Aboamer, M.A.; Sikkandar, M.Y.; Gupta, S.; Vives, L.; Joshi, K.; Omarov, B.; Singh, S.K. An Investigation in Analyzing the Food Quality Well-Being for Lung Cancer Using Blockchain through CNN. *J. Food Qual.* **2022**, *2022*, 5845870.
29. Agarwal, C.; Kaur, I.; Yadav, S. Hybrid CNN-SVM Model for Face Mask Detector to Protect from COVID-19. In *Artificial Intelligence on Medical Data*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 419–426.
30. Dibbo, S.V.; Cheung, W.; Vhaduri, S. On-phone CNN model-based implicit authentication to secure IoT wearables. In *The Fifth International Conference on Safety and Security with IoT*; Springer: Berlin/Heidelberg, Germany, 2023; pp. 19–34.
31. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *arXiv* **2015**, arXiv:1512.03385.
32. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. *arXiv* **2016**, arXiv:1602.07261.
33. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going Deeper with Convolutions. *arXiv* **2014**, arXiv:1409.4842.
34. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. *arXiv* **2015**, arXiv:1512.00567.
35. Li, H. *Exploring Knowledge Distillation of Deep Neural Networks for Efficient Hardware Solutions*; University Of Stanford: CS230 Course Report; University Of Stanford: Stanford, CA, USA, 2018.
36. O'shea, T.J.; West, N. Radio machine learning dataset generation with gnu radio. In Proceedings of the GNU Radio Conference, Charlotte, NC, USA, 20–24 September 2016; Volume 1.
37. Xu, B.; Wang, W.; Falzon, G.; Kwan, P.; Guo, L.; Sun, Z.; Li, C. Livestock classification and counting in quadcopter aerial images using Mask R-CNN. *Int. J. Remote Sens.* **2020**, *41*, 8121–8142.