applied sciences

MDPI

Article

Attentive Octave Convolutional Capsule Network for Medical Image Classification

Hong Zhang [1], Zhengzhen Li [1], Hao Zhao [2], Zan Li [1] and Yanping Zhang [3,*]

[1] School of Computer Science and Technology, Minzu University of China, Beijing 100081, China; zhanghong751103@muc.edu.cn (H.Z.); 19301533@muc.edu.cn (Z.L.); 20301827@muc.edu.cn (Z.L.)
[2] Department of Computer Science and Technology, School of Information, Renmin University of China, Beijing 100872, China; 2019104217@ruc.edu.cn
[3] Department of Computer Science, School of Engineering and Applied Science, Gonzaga University, Spokane, WA 99258, USA
* Correspondence: zhangy@gonzaga.edu; Tel.: +1-509-313-5705

Abstract: Medical image classification plays an essential role in disease diagnosis and clinical treatment. More and more research efforts have been dedicated to the design of effective methods for medical image classification. As an effective framework, the capsule network (CapsNet) can realize translation equivariance. Lots of current research applies capsule networks in medical image analysis. In this paper, we propose an attentive octave convolutional capsule network (AOC-Caps) for medical image classification. In AOC-Caps, an AOC module is used to replace the traditional convolution operation. The purpose of the AOC module is to process and fuse the high- and low-frequency information in the input image simultaneously, and weigh the important parts automatically. Following the AOC module, a matrix capsule is used and the expectation maximization (EM) algorithm is applied to update the routing weights. The proposed AOC-Caps and comparative methods are tested on seven datasets, including PathMNIST, DermaMNIST, OCTMNIST, PneumoniaMNIST, OrganMNIST_Axial, OrganMNIST_Coronal, and OrganMNIST_Sagittal, which are from MedMNIST. In the experiments, baselines include the traditional CNN models, automated machine learning (AutoML) methods, and related capsule network methods. The experimental results demonstrate that the proposed AOC-Caps achieves better performance on most of the seven medical image datasets.

Keywords: medical image classification; capsule network; octave convolution; attention mechanism

Citation: Zhang, H.; Li, Z.; Zhao, H.; Li, Z.; Zhang, Y. Attentive Octave Convolutional Capsule Network for Medical Image Classification. *Appl. Sci.* **2022**, *12*, 2634. https://doi.org/10.3390/app12052634

Academic Editors: Dick Sterenborg and Francesco Bianconi

Received: 14 January 2022
Accepted: 25 February 2022
Published: 3 March 2022

1. Introduction

As an interdisciplinary area, medical image classification is the foundation of automatic disease diagnosis. With the development of deep learning technology, convolutional neural networks (CNNs) [1–7] have been widely applied in computer vision tasks, such as image classification [8,9], object detection [10], semantic segmentation [11], etc. The performance of these tasks was greatly improved with the application of CNNs. However, CNNs have limitations. Firstly, the pooling operation provides some transition invariance and results in the loss of important location information. Secondly, CNNs struggle to learn the part–whole relationship. To address these weaknesses, CapsNet [12] is proposed to replace the scalar output with vector output for representing different properties, such as the orientation and viewpoints of objects. Different from the translation invariance from the pooling operation, CapsNet can provide translation equivariance. Equivariance is the detection of objects that can transform into each other. Different from CNNs, the knowledge about part–whole relationships is kept in the capsule network, as discussed in [12,13]. Capsule networks recognize objects through both local features and part–whole knowledge. For example, a bird, as an object, has several parts, including a head, a trunk, wings, claws, and a tail. When these parts are disturbed, CNNs would still recognize the disturbed object as a bird, while CapsNet can determine that it is not a bird through the part–whole

relationship. In the original CapsNet [12], information is represented in vectorized format, leading to costly calculation of the routing between capsules of different layers. In the matrix CapsNet [13], vectorized information is replaced by matrix capsules, and routing weights are updated by the expectation-maximization (EM) algorithm.

However, early CapsNets have their drawbacks. First, the low-level features that consist of the capsule are extracted only by shallow convolutional operations. This results in capsules containing very little semantic high-level information. Secondly, low-level convolutional operations lack an attention mechanism, which may import meaningless and redundant information into the capsules. One of the effective ways for performance boosting is to employ a better feature extractor, which can capture richer and more semantic contextual patterns to build capsules. Recent efforts focusing on the improvement of feature extraction for CapsNets were extensively investigated, such as Multi-Scale CapsNet (MS-CapsNet) [14] and RS-CapsNet [15].

In this paper, we propose a novel capsule network, named the attentive octave convolutional capsule network (AOC-CapsNet) for medical image classification. In AOC-CapsNet, the traditional convolution operation is replaced by an octave convolution operation. In [16], the octave convolution operation is proposed to process both higher and lower frequencies in the inputs at the same time. In natural images, higher frequencies correspond to the detailed information that varies greatly in the images, and lower frequencies correspond to the smoothly changing structure in the images. These two types of information are also very important for medical image classification. It is critical to select which kind of information is more important in medical image classification. However, the traditional octave convolutional operation cannot enhance useful information and suppress useless information. In AOC-CapsNet, we adopt a convolutional block attention module (CBAM) to identify and select useful information. The CBAM allows AOC-CapsNet to highlight critical local regions with rich semantic details utilized as distinguishable patterns, leading to a performance gain in the medical image classification task.

Studies on capsule networks [17,18] have focused on medical image analysis. A recent benchmark, named MedMNIST [19], was proposed and used to validate the performance of different models for medical image analysis. MedMNIST is composed of 10 pre-processed open medical image datasets. Similar to the MNIST dataset [20], classification tasks in MedMNIST are lightweight. The resolution of images in classification tasks is 28×28. Those tasks cover primary medical image modalities and diverse data scales. In this paper, we design comparative experiments on seven datasets in MedMNIST. Through experiments, ResNet [6], AutoML methods [21,22] and methods related to capsules [12–14] are compared with the proposed AOC-Caps. In the ablation studies, matrix capsule networks with different convolutional feature extraction layers are compared to determine which type of convolution layer is more suitable for the application of capsule networks in the medical image classification of MedMNIST.

The main contributions of this research are as follows:

- An attentive octave convolution operation is proposed. By combining the novel operation with capsule networks, we design an effective classification framework named AOC-CapsNet for medical image classification.
- The proposed AOC-CapsNet is validated via extensive experiments on the MedMNIST benchmark and has achieved the state-of-the-art (SOTA) performance in two of the seven tasks.
- The proposed method can serve as a credible benchmark for future reference. We have made the code public at the following link, which was last accessed on 23 Feburary 2022, https://github.com/aszx87414/Attentive_Octave_Convolutional_Capsule_Network.

The rest of this paper is organized as follows. Section 2 reviews the related research work. Section 3 explains our proposed method. In Section 4, comprehensive experiments are conducted to evaluate the effectiveness of the proposed method. Finally, in Section 5, we conclude the paper.

2. Related Work

Introduced by Hinton [23], the core idea of "capsule" is to group the neurons into a vector, which is defined as a capsule. In CNNs, the activation of a neuron can be considered the likelihood of detecting a specific feature. Different from feature invariance in CNN, feature equivariance, which is considered the detection of features that can transform into each other, is achieved in capsule networks.

In [12], the dynamic routing between capsules is applied in the proposed capsule network. The pooling operation is abandoned in [12] for keeping the location information of features. Although CapsNet with dynamic routing achieved SOTA performance in MNIST and its variant, MultiMNIST, it still has drawbacks, such as huge computational cost and the lack of high-level semantic information. In [13], the matrix capsule network is constructed by transforming the capsule form from vector to matrix and changing the link mode between capsules of different layers. The coupling coefficients between lower-layer and higher-layer capsules are updated by the EM algorithm. In [12,13], all the features used to construct capsules are extracted by a convolution layer. These features are low-level information and cannot effectively recognize complex objects.

To handle the drawbacks explained above, there have been several studies [14,15,17,24,25] focusing on applying more powerful feature extraction modules to improve the performance of capsule networks. In [14], multiple convolutional kernels are used to extract multi-scale features for constructing multi-dimension capsules, and a novel dropout for capsules is proposed. In RS-Caps [15], the Res2Net block [26] is used to extract multi-scale features and increase the size of receptive fields of each convolutional layer. What is more, a new linear combination between capsules and routing process is proposed for constructing more effective classification capsules. In HitNet [24], a new layer called hit-or-miss and a centripetal loss function are designed. HitNet also introduces a data augmentation method that can combine data space and feature space. The most straightforward idea to improve the performance of capsule networks is to increase the number of intermediate capsule layers to obtain deeper capsule networks. However, it was recently proven that directly stacking fully connected capsule layers will result in a decline in performance [27]. In order to solve this problem, DeepCaps [25] uses a novel 3D convolution-based dynamic routing algorithm. Furthermore, a class-independent decoder network is also proposed to strengthen the use of reconstruction loss as a regularization term.

Deep learning technology has also been applied in medical image analysis. In [28], U-Net architecture, which consists of a contracting path to capture context information and an expanding path that enables precise localization, is proposed for biomedical image segmentation. In [29], an approach based on a volumetric, fully convolutional neural network is proposed for 3D image segmentation. USE-Net [30], which incorporates squeeze-and-excitation (SE) modules into U-Net, is proposed for magnetic resonance imaging (MRI) segmentation. In [31], SegNet [32], which consists of an encoder network, a decoder network followed by a pixel-wise classification layer, U-Net and pix2pix are compared in the experiments on two multi-centric MRI prostate datasets.

3. Attentive Octave Convolutional Capsule Network

In this section, we introduce our proposed AOC-Caps in detail. As shown in Figure 1b, input images are fed into a traditional convolution layer followed by batch normalization and RELU operation. The feature maps generated by this convolutional layer are then fed into the attentive octave convolution layer (AOC-Layer). In the AOC-Layer, the higher- and lower-frequencies are processed simultaneously. The useful information is enhanced, and useless information is suppressed in the AOC-Layer. The enhanced feature maps generated by the AOC-Layer are then reshaped into a pose matrix and an activation following the matrix capsule network [13]. In Section 3.1, the details of the AOC-Layer are provided. The process of routing and updating in capsule layers is introduced in Section 3.2. The loss function of the proposed AOC-Caps is described in Section 3.3.

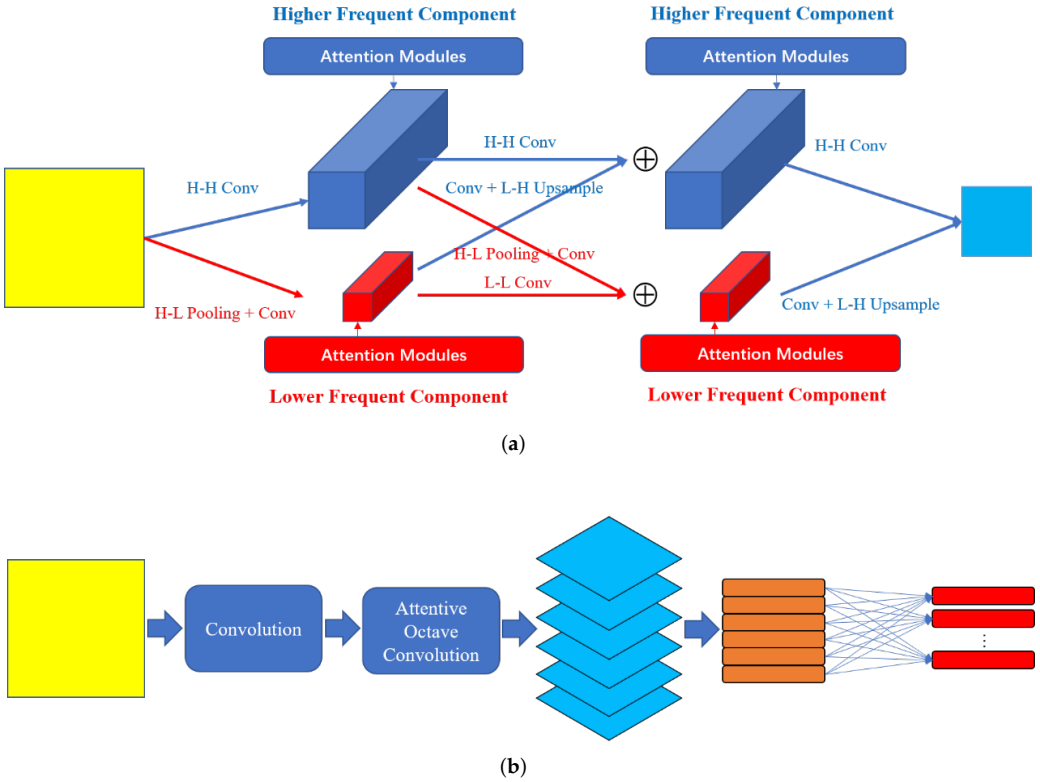

Figure 1. Attentive octave convolution capsule network. (**a**) Attentive octave convolution layer. (**b**) Main framework of AOC-Caps.

3.1. Attentive Octave Convolution Layer

In the traditional convolution operation, the input information is processed by convolutional kernels of certain sizes. Convolution operations of different convolutional kernels can obtain information of different frequencies, and there is no effective fusion process between frequencies. The octave convolution operation [16] is proposed to process different frequencies simultaneously. In octave convolution, the convolution and fusion of two frequencies with a difference of an octave are performed simultaneously without an attention mechanism module. It is very important to select useful information in medical image classification. In order to select information of different frequencies, we add a CBAM [33] in the AOC-Layer.

Suppose the input is defined as $I \in R^{3\times h\times w}$, where h and w are defined as the height and width of the image. The feature maps obtained by the first convolution layer are defined as $F \in R^{c\times h\times w}$, where c is the channel of the feature maps. In the AOC-Layer, the feature map F is first divided into two parts $\{F_1^H, F_1^L\}$ by a convolution operation (H-H Conv in Figure 1a) and pooling and a convolution operation (H-L Pooling and Conv in Figure 1a), where F_1^H is higher frequency and F_1^L is lower frequency. The channels of the feature maps are divided by ratio α. The size of the higher frequency is $R^{\alpha c\times h\times w}$ and that of the lower frequency is $R^{(1-\alpha)c\times \frac{h}{2}\times \frac{w}{2}}$. In order to obtain lower-frequency information, two pooling operations (average and maximum) are used in the AOC-Layer. Their effects are discussed in detail in Section 4.5. In order to convert lower-frequency information into higher-frequency information, bilinear interpolation is used to convert lower-resolution feature maps into higher-resolution feature maps. Then, the higher- and lower-frequency communicate with each other by summation, as shown in Formulas (1) and (2):

$$F_2^H = Conv_{H-H}(F_1^H) + Upsample_{L-H}(Conv_{L-H}(F_1^L)) \quad (1)$$

$$F_2^L = Pooling_{H-L}(Conv_{H-L}(F_1^H)) + Conv_{L-L}(F_1^L) \quad (2)$$

As shown in Figure 1a, the attention modules are added to the intermediate feature maps with different frequencies to enhance useful information and suppress useless information. Without losing particularity, an intermediate feature map in the AOC-Layer is defined as $F^{'} \in R^{C^{'} \times H^{'} \times W^{'}}$, where $C^{'}$ is the number of channels of both higher- and lower-frequency parts, and $H^{'}$ and $W^{'}$ are the height and width of higher- and lower-frequency parts. The attention module sequentially infers a 1D channel attention map $F_c^{'} \in R^{C^{'} \times 1 \times 1}$ and a 2D spatial attention map $F_s^{'} \in R^{1 \times H^{'} \times W^{'}}$. The selections of channel and spatial information are based on Formulas (3) and (4):

$$F_c^{att} = F_c^{'}(F^{'}) \otimes F^{'} \tag{3}$$

$$F_{s,c}^{att} = F_s^{'}(F_c^{att}) \otimes F_c^{att} \tag{4}$$

where $\otimes$ denotes the element-wise multiplication. The intermediate feature map $F^{'} \in R^{C^{'} \times H^{'} \times W^{'}}$ is firstly processed by a channel attention module. The feature map $F^{'}$ is pooled along the spatial dimension through average and maximum operations. The average-pooled features and max-pooled features are processed by a multi-layer perception (MLP), with one hidden layer for producing the channel attention vector $F_c^{'} \in R^{C^{'} \times 1 \times 1}$. The channel attention enhanced feature map F_c^{att} is then processed by a spatial attention module. Feature map F_c^{att} is pooled along the channel dimension by both average and maximum operations. The average-pooled and max-pooled feature maps are concatenated along the channel dimension to produce $F_s^{'} \in R^{2 \times H^{'} \times W^{'}}$. $F_s^{'}$ is processed by a convolution operation with kernel size 7 followed by a sigmoid function. In the AOC-Layer, the attention modules are plugged into F_1^H, F_1^L, F_2^H, and F_2^L. The role of attention modules is also discussed in detail in Section 4.5.

3.2. Capsule Layer

The two commonly used capsule networks are CapsNet with dynamic routing and matrix CapsNet with EM routing. In CapsNet with dynamic routing, the capsule vector is constructed by stacking the neurons with scalar values. In CapsNet with EM routing, the capsule contains a pose matrix and an activation.

The output feature maps of the AOC-Layer are first reshaped into a series of capsules $s_j, j = 1, 2, \cdots$. For one capsule s_j, its input is the weighted sum of all the prediction vectors $\hat{u}_{j|i}$ generated by the previous layers. It can be defined as Formulas (5) and (6).

$$s_j = \sum_i c_{ij} \hat{u}_{j|i} \tag{5}$$

$$\hat{u}_{j|i} = W_{ij} u_i \tag{6}$$

where c_{ij} is the routing coefficient, W_{ij} is the matrix used for voting, and u_i is the output vector of the previous capsule layer. The routing coefficients should be computed in dynamic routing [12] or EM routing [13]. In dynamic routing, u_i is a vector.

The process of dynamic routing is shown as follows:

1. The prior probability b_{ij} between capsule j and capsule i in the previous layer is initialized to be 0;
2. The routing coefficients c_{ij} can be computed through the softmax function $c_{ij} = \frac{exp(b_{ij}))}{\sum_k exp(b_{ik})}$;
3. The input to capsule j is computed by Formula (5) and then it is squeezed by $v_j = \frac{\|s_j\|^2}{1+\|s_j\|^2} \frac{s_j}{\|s_j\|}$;
4. The b_{ij} is updated by $b_{ij} \leftarrow b_{ij} + \hat{u}_{j|i} \cdot v_j$;
5. Repeat steps 2 to 4 r times. The value of r is set empirically, usually from 1 to 3.

Different from capsule networks with dynamic routing, capsules in the matrix capsule with EM routing consist of a pose matrix and an activation. A pose matrix defines the translation and the rotation of the objects. The aim of the EM algorithm is to cluster datapoints into different Gaussian distributions. Suppose the pose matrix is a 4×4 matrix, i.e., 16 components. Let v_{ij} be the vote from capsule i to capsule j, and v_{ij}^h be its h-th component. The probability density function of a Gaussian is defined as Formula (7):

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{7}$$

It can be applied to compute the probability of v_{ij}^h belonging to the capsule j's Gaussian model:

$$P_{i|j}^h = \frac{1}{\sqrt{2\pi\left(\sigma_j^h\right)^2}} e^{-\frac{\left(v_{ij}^h - \mu_j^h\right)^2}{2\left(\sigma_j^h\right)^2}} \tag{8}$$

Let $cost_{ij}^h = -ln\left(P_{i|j}^h\right)$ be the cost to activate the h-th component of capsule j by the h-th component of capsule i, where

$$ln\left(P_{i|j}^h\right) = -ln(\sigma_j^h)) - \frac{ln(2\pi)}{2} - \frac{\left(v_{ij}^h - \mu_j^h\right)^2}{2\left(\sigma_j^h\right)^2} \tag{9}$$

Whether the capsule j is activated is determined by the following equation:

$$a_j = sigmoid(\lambda(b_j - \sum_h cost_j^h)) \tag{10}$$

where

$$cost_j^h = \sum_i r_{ij} cost_{ij}^h \tag{11}$$

r_{ij} is the assignment probability of each datapoint to a capsule. r_{ij}, μ, σ, and a_j are computed iteratively using the EM algorithm.

3.3. Loss Function

In AOC-Caps, if the dynamic routing is applied, the loss function is defined as

$$L_c = T_c max(0, m^+ - \|v_c\|)^2 + \lambda(1 - T_c) max(0, \|v_c\| - m^-)^2 \tag{12}$$

where $T_c = 1$ if the class c is present, m^+ is set to be 0.9 and m^- is set to be 0.1.

If the matrix capsule with EM routing is applied, the loss function has a similar design as in [13]. The spread loss is used to directly maximize the gap between the activation of the target class and the activation of other classes. The loss function is formed as Formulas (13) and (14):

$$L_w = max(0, m - (a_r - a_w))^2 \tag{13}$$

$$L = \sum_{w \neq r} L_w \tag{14}$$

where a_w is the wrong class and a_t is the target class.

4. Experiments

4.1. Datasets

MedMNIST consists of 10 pre-processed datasets. It contains 10 open medical image datasets covering multiple tasks, including multi-class, binary classification, sequential

regression, and multi-label. In our experiments, we focus on the multi-class tasks, such as PathMNIST, DermaMNIST, OCTMNIST, PneumoniaMNIST, OrganMNIST_Axial, OrganMNIST_Coronal, and OrganMNIST_Sagittal. In these seven datasets, the height and width of the images are resized to 28. Figure 2 demonstrates an overview of the seven datasets with samples.

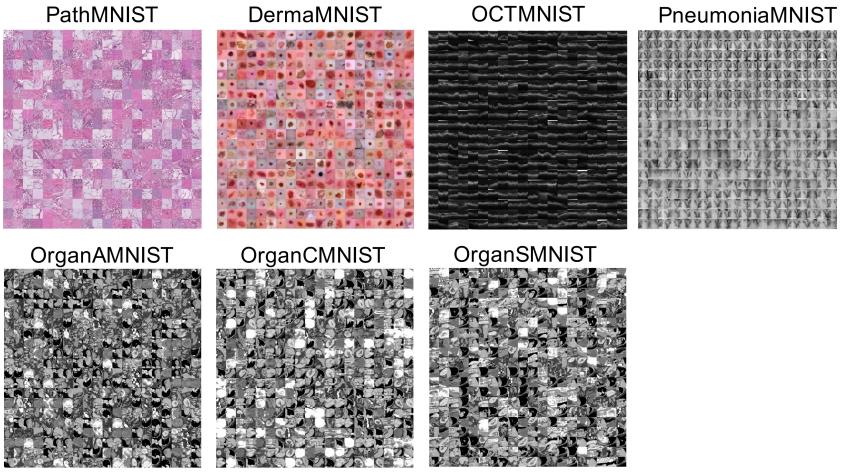

Figure 2. An Overview of the Datasets: PathMNIST, DermaMNIST, OCTMNIST, PneumoniaMNIST, OrganMNIST_Axial, OrganMNIST_Coronal, and OrganMNIST_ Sagittal.

All datasets are divided into a training set, a validation set, and a test set. The number of images in each set is detailed in Table 1. The models are trained on the training sets, validated on the validation sets after each epoch during training, and finally evaluated on the test sets.

Table 1. Datasets in MedMNIST used in our experiments.

Name	Data Modality	Tasks	#Training	#Validation	#Test
PathMNIST	Pathology	Multi-Class	89,996	10,004	7180
DermaMNIST	Dermatoscope	Multi-Class	7007	1003	2005
OCTMNIST	OCT	Multi-Class	97,477	10,832	1000
PneumoniaMNIST	Chest X-ray	Binary-Class	4708	524	624
OrganMNIST_Axial	Abdominal CT	Multi-Class	34,581	6491	17,778
OrganMNIST_Coronal	Abdominal CT	Multi-Class	13,000	2392	8268
OrganMNIST_Sagittal	Abdominal CT	Multi-Class	13,940	2452	8829

PathMNIST. It is based on a prior study [34] for predicting survival from colorectal cancer histology slides, which provides a dataset of 100,000 non-overlapping image patches from hematoxylin and eosin-stained histological images, and a test dataset of 7180 images patches from a different clinical center. Nine types of tissues are involved, resulting in a multi-class classification task. The details of these nine categories are introduced in Table 2.

Table 2. Details of PathMNIST dataset.

Name	#Training	#Validation	#Testing
adipose	9366	1041	1338
background	9509	1057	847
debris	10,360	1152	339
lymphocytes	10,401	1156	634
mucus	8006	890	1035
smooth muscle	12,182	1354	592
normal colon mucosa	7886	877	741
cancer-associated stroma	9401	1045	421
colorectal adenocarcinoma epithelium	12,885	1432	1233

DermaMNIST. It is based on HAM10000 [35], a large collection of multi-source dermatoscopic images of common pigmented skin lesions. The dataset consists of 10,015 images labeled as seven different categories, as a multi-class classification task. These seven categories are introduced in Table 3.

Table 3. Details of DermaMNIST dataset.

Name	#Training	#Validation	#Testing
Actinic Keratoses and Intraepithelial Carcinoma	228	33	66
Basal Cell Carcinoma	359	52	103
Benign Keratosis-like Lesions	769	110	220
Dermatofibroma	80	12	23
Melanoma	779	111	223
Melanocytic Nevi	4693	671	1341
Vascular Lesions	99	14	29

OctMNIST. It is based on a prior dataset [36] of 109,309 valid optical coherence tomography images for retinal diseases. Four types are involved in this dataset, leading to a multi-class classification task. These four categories are introduced in Table 4.

Table 4. Details of OctMNIST dataset.

Name	#Training	#Validation	#Testing
Choroidal Neovascularization	33,484	3721	250
Diabetic Macular Edema	10,213	1135	250
Drusen	7754	862	250
Normal	46,026	5114	250

PneumoniaMNIST. It is based on a prior dataset [36] of 5856 pediatric chest X-ray images. This task is a binary class of pneumonia and normal. The source training set is split into training and validation sets with a ratio of 9:1, and its source validation set is used as the test set. These two categories are introduced in Table 5.

Table 5. Details of PneumoniaMNIST dataset.

Name	#Training	#Validation	#Testing
Normal	1214	135	234
Pneumonia	3494	389	390

OrganMNIST_(Axial, Coronal, and Sagittal). These three datasets are based on 3D computed tomography (CT) images from the Liver Tumor Segmentation Benchmark [37]. Bounding-box annotations of 11 body organs from another study are used for obtaining the organ labels. The only differences of OrganMNIST_(Axial, Coronal, and Sagittal) are the views. The 11 categories of each of the three datasets are introduced in Tables 6–8.

Table 6. Details of OrganMNIST_Axial dataset.

Name	#Training	#Validation	#Testing
Bladder	1956	321	1036
Femur-left	1408	233	784
Femur-right	1359	225	793
Heart	1474	392	785
Kidney-left	3963	568	2064
Kidney-right	3817	637	1965
Liver	6164	1033	3285
Lung-left	3919	1033	1747
Lung-right	3929	1009	1813
Pancreas	3031	529	1622
Spleen	3561	511	1884

Table 7. Details of OrganMNIST_Coronal dataset.

Name	#Training	#Validation	#Testing
Bladder	1153	191	833
Femur-left	626	102	442
Femur-right	608	96	441
Heart	600	202	421
Kidney-left	1088	132	732
Kidney-right	1170	157	737
Liver	2986	429	1836
Lung-left	1002	347	550
Lung-right	1022	352	558
Pancreas	1173	179	750
Spleen	1572	205	968

Table 8. Details of OrganMNIST_Sagittal dataset.

Name	#Training	#Validation	#Testing
Bladder	1148	188	811
Femur-left	637	104	439
Femur-right	615	95	447
Heart	721	246	510
Kidney-left	1132	140	704
Kidney-right	1119	159	693
Liver	3464	491	2078
Lung-left	741	261	397
Lung-right	803	275	439
Pancreas	2004	280	1343
Spleen	1556	213	968

4.2. Evaluation Metrics

In our experiments, we use accuracy (ACC), area under ROC curve (AUC), precision (PRE), recall (REC) and F1-score (F1) as the evaluation metrics. The formulas of these metrics are shown below:

$$Accurary = \frac{\sum_{i=1}^{N} f(x_i) = y_i}{N} \tag{15}$$

$$precision = \frac{\sum_{i=1}^{c} precision_i}{C} \tag{16}$$

$$precision_i = \frac{TP_i}{TP_i + FP_i}, i = 1, 2, \ldots, C \tag{17}$$

$$recall = \frac{\sum_{i=1}^{C} recall_i}{C} \tag{18}$$

$$recall_i = \frac{TP_i}{TP_i + FN_i}, i = 1, 2, \ldots, C \tag{19}$$

$$\mathrm{F1} = \frac{2 * precision * recall}{precision + recall} \tag{20}$$

where C is the number of classes and N is the number of total samples. TP, TN, FP, and FN refer to true positive, true negative, false positive, and false negative.

Accuracy (ACC) is the most commonly used metric among these performance metrics, but it does not indicate the true model performance when the classes are imbalanced. Area under the ROC curve (AUC) is less sensitive to class imbalance than ACC. Precision (PRE) and recall (REC) are related to the positive prediction. In our experiments, we use macro-precision and macro-recall, which are defined by Formulas (16) and (18). The F1-score (F1) is a metric that combines both precision and recall.

4.3. Baselines

In our experiments, we use the same baselines as in [19]. In addition, several methods related to capsule networks are used in the classification tasks of the seven datasets mentioned above.

ResNet18 and ResNet50 [6]. These two models are trained for 100 epochs, using a cross-entropy loss function and an SGD optimizer with a bath size of 128 and an initial learning rate 10^{-3}.

AutoML Methods. Several AutoML methods [21,22] were applied on MedMNIST classification. The experimental settings of three AutoML methods (auto-sklearn [21], AutoKeras [22] and Google AutoML Vision) are the same as in [19]. AutoML methods are designed to search for the optimal hyper-parameter setting or neural architecture to maximize the predictive ability. For example, Auto-sklearn and autoKeras are open-source AutoML tools for both statistical machine learning and deep learning. On the other hand, Google AutoML Vision offers commercial black-box AutoML tools. In this study, the results of AutoML methods in our experiments are directly referenced from [19].

CapsNet (Dynamic routing) [12]. In [12], the output feature maps of the convolutional layer with kernel size 9×9 and a stride 2 are reshaped into primary capsules. The capsules of the previous layer are routed to classification capsules by agreement. Different from the original setting, we set the iteration number to be 2 instead of 3 for the best performance.

CapsNet (EM routing) [13]. In [13], the capsule contains a 4×4 pose matrix and an activation. The vote for capsules in the next layer is computed by the matrix multiplication between the pose matrix and the trainable transformation matrix. The routing coefficients between capsules and classification capsules are updated through the EM algorithms. Different from the original setting, the kernel size of the first convolutional layer is set to be 3×3 for detailed feature representation.

MS-Caps [14]. In [14], hierarchical features are extracted and reshaped into capsules of different dimensions. The capsules cascade together for the dropout operation. Following the original experimental setting, the Adam optimizing method [38] is used as the gradient descent algorithm to perform the training. The weight decay factor is set to be 0.00001. The initial learning rate is 0.001 and 0.0001, and the number of iterations is 25 and 50 for converging to the optimal solution quickly.

DeepCaps [25]. In [25], the Adam optimizer is used with an initial learning rate of 0.001, which is reduced by half after every 20 epochs.

4.4. Implementation Details

In this paper, experiments are implemented by PyTorch on a PC with four GPUs of TITAN X. Different from the experiments in [19], we conduct experiments only with images of size 28×28.

In the experiments, the kernel size of the first convolution layer is set to be 3×3 and the stride is set to be 2. In the AOC-Layer, ratios for the split of higher- and lower-frequencies are set to be 0.5. The convolutional kernel size is set to be 3 in the AOC-Layer. In the attention modules, the reduction ratio is set to be 8, and the convolutional kernel size is set to be 3. When the capsule with a dynamic routing framework is applied, the vector dimension of the capsule is set to be 8 and the iteration number is set to be 2. When the matrix capsule with the EM routing framework is applied, the pose matrix is set to be a 4×4 matrix. The batch size is set to be 96 and the initial learning rate is 3×10^{-3}. The training epoch is set to be 120.

In our experiments, all metrics are implemented by a python module called Torchmetrics.

4.5. Ablation Study

To verify the effectiveness of each design in AOC-Caps, we conduct experiments for the ablation study. The ablation study focuses on (1) different forms of capsules and different routing methods; (2) types of pooling operations in the AOC-Layer; (3) the effect of attention modules.

To test different forms of capsules and different routing methods, we choose the pooling operation type to be maximal and plug attention modules as introduced in Section 3.1. It should be noted that in dynamic routing, we do not add the decode term. As shown in Table 9, the AOC-Caps with matrix and EM routing outperforms the one with dynamic routing. It can be seen from Table 9 that different routing methods have a great impact on the results. For example, in the OrganMNIST_Axial dataset, the accuracy with EM routing

is 93.1%, while the accuracy with dynamic routing is 80.2%. The AOC-Caps with EM routing also achieves better precision and recall, both of which are more than 10% higher.

Table 9. Performance of different capsule forms and routing.

Dataset Names	AOC-Caps (Dynamic Routing)					AOC-Caps (EM Routing)				
	ACC	PRE	REC	F1	AUC	ACC	PRE	REC	F1	AUC
PathMNIST	0.794	0.790	0.782	0.783	0.952	**0.859**	**0.840**	**0.837**	**0.845**	**0.960**
DermaMNIST	0.732	0.420	0.368	0.390	0.883	**0.786**	**0.485**	**0.417**	**0.447**	**0.890**
OCTMNIST	0.707	0.788	0.760	0.773	0.950	**0.750**	**0.820**	**0.802**	**0.819**	**0.955**
PneumoniaMNIST	0.859	0.862	0.843	0.847	0.985	**0.931**	**0.914**	**0.893**	**0.895**	**0.990**
OrganMNIST_Axial	0.802	0.799	0.773	0.785	0.950	**0.931**	**0.920**	**0.903**	**0.919**	**0.973**
OrganMNIST_Coronal	0.794	0.803	0.892	0.899	0.940	**0.907**	**0.911**	**0.904**	**0.904**	**0.965**
OrganMNIST_Sagittal	0.739	0.680	0.665	0.669	0.967	**0.783**	**0.731**	**0.717**	**0.723**	**0.980**

To test the effectiveness of different pooling operations, such as average and maximum, we keep the capsule form to matrix and EM routing. The attention modules are plugged into the AOC-Layer as introduced in Section 3.1. As shown in Table 10, the AOC-Caps with max-pooling outperforms the one with avg-pooling. What is more, the type of pooling operations has much less impact on the results than the routing methods.

Table 10. Performance of different pooling operations in AOC-Layer.

Dataset Names	AOC-Caps (AVG)					AOC-Caps (MAX)				
	ACC	PRE	REC	F1	AUC	ACC	PRE	REC	F1	AUC
PathMNIST	0.847	0.829	0.820	0.832	0.960	**0.859**	**0.840**	**0.837**	**0.845**	**0.960**
DermaMNIST	0.764	0.447	0.380	0.403	0.860	**0.786**	**0.485**	**0.417**	**0.447**	**0.890**
OCTMNIST	0.739	0.807	0.789	0.798	0.930	**0.750**	**0.820**	**0.802**	**0.819**	**0.955**
PneumoniaMNIST	0.918	0.895	0.879	0.880	0.985	**0.931**	**0.914**	**0.893**	**0.895**	**0.990**
OrganMNIST_Axial	0.927	0.915	0.900	0.912	0.970	**0.931**	**0.920**	**0.903**	**0.919**	**0.973**
OrganMNIST_Coronal	0.886	0.889	0.878	0.883	0.960	**0.907**	**0.911**	**0.904**	**0.904**	**0.965**
OrganMNIST_Sagittal	0.769	0.718	0.702	0.710	0.970	**0.783**	**0.731**	**0.717**	**0.723**	**0.980**

To test the effectiveness of the attention modules, we keep the capsule form to matrix and EM routing. The pooling type is set to be maximal. As shown in Table 11, the AOC-Caps with the attention module outperforms the one without the attention module. Attention modules have a greater impact on the results than the pooling operation type and less than the routing method. In Table 11, the AOC-Caps with attention modules achieves higher accuracy, precision and recall.

Table 11. Performance of AOC-Caps with/without attention module.

Dataset Names	AOC-Caps (without Attention)					AOC-Caps (with Attention)				
	ACC	PRE	REC	F1	AUC	ACC	PRE	REC	F1	AUC
PathMNIST	0.805	0.790	0.785	0.785	0.945	**0.859**	**0.840**	**0.837**	**0.845**	**0.960**
DermaMNIST	0.721	0.405	0.337	0.382	0.850	**0.786**	**0.485**	**0.417**	**0.447**	**0.890**
OCTMNIST	0.689	0.756	0.743	0.749	0.943	**0.750**	**0.820**	**0.802**	**0.819**	**0.955**
PneumoniaMNIST	0.863	0.847	0.826	0.833	0.980	**0.931**	**0.914**	**0.893**	**0.895**	**0.990**
OrganMNIST_Axial	0.877	0.864	0.845	0.856	0.972	**0.931**	**0.920**	**0.903**	**0.919**	**0.973**
OrganMNIST_Coronal	0.832	0.843	0.830	0.836	0.955	**0.907**	**0.911**	**0.904**	**0.904**	**0.965**
OrganMNIST_Sagittal	0.711	0.654	0.630	0.648	0.955	**0.783**	**0.731**	**0.717**	**0.723**	**0.980**

The results in Tables 9–11 have demonstrated the effectiveness of EM routing, max-pooling in the AOC-Layer, and attention modules. By putting them together, we find the optimal combination to implement AOC-Caps based on the following comparative experiments.

4.6. Comparative Experiments

The results of the comparative experiments are reported in Table 12, in which the results of ResNet18, ResNet50, Auto-sklearn, AutoKeras, and Google AutoML Vision are from [19]. For the three AutoML models, the original paper does not provide metrics in Pre, Rec, and F1, which are marked as dashes in Table 12.

Table 12 shows the performance of the comparative models on seven datasets in MedMNIST. In terms of accuracy (ACC), AOC-Caps achieves the best accuracy on DermaMNIST and OrganMNIST_Axial datasets, and ranks second on other datasets. CapsNet with dynamic routing and EM routing demonstrate worse performance than other methods, due to the shallow feature extraction network. However, because the data of OrganMNIST_(Axial, Coronal, and Sagittal) are collected from 3D images, the viewpoints are different. Therefore, the CapsNet with the EM routing with transformation invariance can obtain good accuracy, even with the shallow feature extraction network.

Due to the class imbalance in certain datasets of MedMNIST, it is necessary to consider different metrics. In terms of precision (PRE) and recall (REC), the performance of all methods is lower than accuracy on datasets DermaMNIST and OrganMNIST_Sagittal because of the data imbalance between different categories in DermaMNIST and OrganMNIST_Sagittal. In this case, AOC-Caps still achieves better results. However, it should be pointed out that there is no corresponding solution to this data imbalance problem in our experiments.

In our comparative experiments, a model with deeper layers does not necessarily achieve higher accuracy. Consider ResNet18 as an example. It outperforms ResNet50 on datasets such as DermaMNIST, OCTMNIST, OrganMNIST_Axial, and OrganMNIST_Sagittal. As part of the reason, the low resolution of the image data does not require a deep network model to extract high-level features.

Table 12. Performance of seven datasets in MedMNIST.

Dataset Names	ResNet18					ResNet50				
	ACC	**PRE**	**REC**	**F1**	**AUC**	**ACC**	**PRE**	**REC**	**F1**	**AUC**
PathMNIST	0.844	0.817	0.817	0.805	0.966	**0.864**	0.832	0.833	0.820	0.978
DermaMNIST	0.721	0.468	0.373	0.391	0.895	0.710	0.363	0.332	0.343	0.886
OCTMNIST	**0.758**	0.814	0.749	0.713	0.951	0.745	0.809	0.730	0.702	0.951
PneumoniaMNIST	0.843	0.895	0.797	0.817	0.970	0.857	0.895	0.797	0.817	0.966
OrganMNIST_Axial	0.921	**0.924**	**0.912**	0.917	0.995	0.916	0.917	0.906	0.910	0.995
OrganMNIST_Coronal	0.889	0.889	0.886	0.886	0.990	0.893	0.890	0.886	0.886	**0.992**
OrganMNIST_Sagittal	0.762	0.714	0.706	0.694	0.969	0.746	0.705	0.697	0.692	0.970
Dataset Names	**Auto-sklearn**					**AutoKeras**				
	ACC	**PRE**	**REC**	**F1**	**AUC**	**ACC**	**PRE**	**REC**	**F1**	**AUC**
PathMNIST	0.186	_	_	_	0.500	**0.864**	_	_	_	0.979
DermaMNIST	0.734	_	_	_	0.906	0.756	_	_	_	0.921
OCTMNIST	0.595	_	_	_	0.883	0.736	_	_	_	0.956
PneumoniaMNIST	0.865	_	_	_	0.947	0.918	_	_	_	0.970
OrganMNIST_Axial	0.563	_	_	_	0.797	0.929	_	_	_	**0.996**
OrganMNIST_Coronal	0.676	_	_	_	0.898	0.915	_	_	_	**0.992**
OrganMNIST_Sagittal	0.601	_	_	_	0.855	**0.803**	_	_	_	0.972
Dataset Names	**Google AutoML Vision**					**CapsNet : Dynamic**				
	ACC	**PRE**	**REC**	**F1**	**AUC**	**ACC**	**PRE**	**REC**	**F1**	**AUC**
PathMNIST	0.811	_	_	_	**0.982**	0.710	0.693	0.665	0.682	0.851
DermaMNIST	0.766	_	_	_	**0.925**	0.601	0.332	0.314	0.308	0.807
OCTMNIST	0.732	_	_	_	**0.965**	0.598	0.657	0.677	0.669	0.890
PneumoniaMNIST	**0.941**	_	_	_	**0.993**	0.738	0.793	0.773	0.779	0.930
OrganMNIST_Axial	0.818	_	_	_	0.988	0.738	0.758	0.741	0.747	0.923
OrganMNIST_Coronal	0.861	_	_	_	0.986	0.740	0.747	0.730	0.742	0.943
OrganMNIST_Sagittal	0.706	_	_	_	0.964	0.635	0.595	0.578	0.583	0.852
Dataset Names	**CapsNe : EM**					**MS-Caps**				
	ACC	**PRE**	**REC**	**F1**	**AUC**	**ACC**	**PRE**	**REC**	**F1**	**AUC**
PathMNIST	0.810	0.799	0.776	0.783	0.951	0.843	0.830	0.809	0.817	0.955
DermaMNIST	0.713	0.430	0.372	0.398	0.867	0.720	0.463	0.389	0.412	0.880
OCTMNIST	0.695	0.775	0.730	0.752	0.932	0.673	0.754	0.730	0.749	0.950
PneumoniaMNIST	0.842	0.879	0.836	0.858	0.960	0.810	0.833	0.819	0.825	0.960
OrganMNIST_Axial	0.870	0.883	0.845	0.859	0.987	0.889	0.890	0.867	0.873	0.980
OrganMNIST_Coronal	0.843	0.857	0.833	0.839	0.980	0.863	0.870	0.851	0.859	0.977
OrganMNIST_Sagittal	0.701	0.667	0.601	0.628	0.963	0.742	0.701	0.678	0.688	0.962
Dataset Names	**DEEPCaps**					**AOC-Caps**				
	ACC	**PRE**	**REC**	**F1**	**AUC**	**ACC**	**PRE**	**REC**	**F1**	**AUC**
PathMNIST	0.791	0.783	0.770	0.779	0.965	0.859	**0.840**	**0.837**	**0.845**	0.960
DermaMNIST	0.749	0.406	0.337	0.390	0.898	**0.786**	**0.485**	**0.417**	**0.447**	0.890
OCTMNIST	0.600	0.676	0.659	0.663	0.920	0.750	**0.820**	**0.802**	**0.819**	0.955
PneumoniaMNIST	0.821	0.847	0.828	0.836	0.955	0.931	**0.914**	**0.893**	**0.895**	0.990
OrganMNIST_Axial	0.856	0.867	0.851	0.859	0.961	**0.931**	0.920	0.903	**0.919**	0.973
OrganMNIST_Coronal	0.847	0.855	0.842	0.843	0.950	**0.907**	**0.911**	**0.904**	**0.904**	0.965
OrganMNIST_Sagittal	0.737	0.693	0.659	0.673	0.960	0.783	**0.731**	**0.717**	**0.723**	**0.980**

5. Conclusions

In this paper, we proposed a novel attentive octave convolutional capsule network (AOC-Caps) for medical image classification. In the AOC-Layer, the octave convolution and attention modules (CBAM) are used for communicating and selecting the higher- and lower-frequency information. The output feature maps of the AOC-Layer are used as the foundation for constructing capsules. The experiments have verified the effectiveness of the AOC-Layer. Because the images of certain datasets come from 3D images and the viewpoint is likely to change, the capsule routing method with transformation invariance, such as matrix CapsNet with the EM routing, can obtain higher accuracy. By combining the AOC-Layer and matrix CapsNet with the EM routing, AOC-Caps could achieve better performance than most baselines in the experiments.

However, the AOC-Layer in AOC-Caps is still a shallow convolutional network. Although AOC-Caps has achieved better results than DeepCaps on the classification tasks in MedMNIST, the deep network structure is still necessary when handling high-resolution medical images. This may be due to the smaller image size of the dataset and less detailed information. In future studies, we will consider high-resolution medical images of different diseases and investigate the effect of AOC-Caps in the case of deep structures. Furthermore, it is necessary to consider how to solve the problem of class imbalance in medical image analysis.

Author Contributions: Conceptualization and methodology, Z.L. (Zhengzhen Li) and H.Z. (Hong Zhang); software, validation, Z.L. (Zan Li) and H.Z. (Hao Zhao); original draft preparation, H.Z. (Hong Zhang) and Y.Z.; review and editing, H.Z. (Hong Zhang), Z.L. (Zhengzhen Li) and Y.Z. 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: Not applicable. The data and code supporting the conclusions of this article are available at (last accessed on 23 Feburary 2022) https://github.com/aszx87414/Attentive_Octave_Convolutional_Capsule_Network.

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

References

1. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet classification with deep convolutional neural networks. In Proceedings of the 25th International Conference on Neural Information Processing Systems, Siem Reap, Cambodia, 13–16 December 2012; Curran Associates Inc.: Lake Tahoe, NV, USA, 2012; Volume 1.
2. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2014**, arXiv:1409.1556
3. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015. [CrossRef]
4. Szegedy, C.; Vanhoucke, V.; Ioffe, S.; Shlens, J.; Wojna, Z. Rethinking the Inception Architecture for Computer Vision. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016; Volume 1, pp. 2818–2826. [CrossRef]
5. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In Proceedings of the Thirty-First AAAI Conference on Artificial IntelligenceFebruary, San Francisco, CA, USA, 4–9 February 2017; pp. 4278–4284.
6. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 27–30 June 2016. [CrossRef]
7. Xie, S.; Girshick, R.; Dollár, P.; Tu, Z.; He, K. Aggregated Residual Transformations for Deep Neural Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017. [CrossRef]
8. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. ImageNet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
9. Zhou, B.; Lapedriza, A.; Khosla, A.; Oliva, A.; Torralba, A. Places: A 10 Million Image Database for Scene Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1452–1464. [CrossRef] [PubMed]

10. 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.* **2017**, *39*, 1137–1149. [CrossRef]
11. Long, J.; Shelhamer, E.; Darrell, T. Fully convolutional networks for semantic segmentation. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 3431–3440. [CrossRef]
12. Sabour, S.; Frosst, N.; Hinton, G.E. Dynamic Routing Between Capsules. *arXiv* **2017**, arXiv:1710.09829.
13. Hinton, G.E.; Sabour, S.; Frosst, N. Matrix capsules with EM routing. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
14. Xiang, C.; Zhang, L.; Tang, Y.; Zou, W.; Xu, C. MS-CapsNet: A Novel Multi-Scale Capsule Network. *IEEE Signal Process. Lett.* **2018**, *25*, 1850–1854. [CrossRef]
15. Yang, S.; Lee, F.; Miao, R.; Cai, J.; Chen, L.; Yao, W.; Kotani, K.; Chen, Q. RS-CapsNet: An Advanced Capsule Network. *IEEE Access* **2020**, *8*, 85007–85018. [CrossRef]
16. Chen, Y.; Fan, H.; Xu, B.; Yan, Z.; Kalantidis, Y.; Rohrbach, M.; Shuicheng, Y.; Feng, J. Drop an Octave: Reducing Spatial Redundancy in Convolutional Neural Networks with Octave Convolution. In Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV), Seoul, Korea, 27–28 October 2019; Volume 1, pp. 3434–3443. [CrossRef]
17. Hoogi, A.; Wilcox, B.; Gupta, Y.; Rubin, D.L. Self-Attention Capsule Networks for Image Classification. *arXiv* **2019**, arXiv:1904.12483
18. LaLonde, R.; Torigian, D.A.; Bagci, U. Encoding High-Level Visual Attributes in Capsules for Explainable Medical Diagnoses. *arXiv* **2019**, arXiv:1909.05926.
19. Yang, J.; Shi, R.; Ni, B. MedMNIST Classification Decathlon: A Lightweight AutoML Benchmark for Medical Image Analysis. *arXiv* **2020**, arXiv:2010.14925.
20. Deng, L. The MNIST Database of Handwritten Digit Images for Machine Learning Research [Best of the Web]. *IEEE Signal Process. Mag.* **2012**, *29*, 141–142. [CrossRef]
21. Feurer, M.; Klein, A.; Eggensperger, K.; Springenberg, J.T.; Blum, M.; Hutter, F. *Auto-Sklearn: Efficient and Robust Automated Machine Learning*; The Springer Series on Challenges in Machine Learning; Springer: Cham, Switzerland, 2019.
22. Jin, H.; Song, Q.; Hu, X. Auto-Keras: An Efficient Neural Architecture Search System. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Anchorage, AK, USA, 4–8 August 2019. [CrossRef]
23. Hinton, G.E.; Krizhevsky, A.; Wang, S.D. *Transforming Auto-Encoders. Artificial Neural Networks and Machine Learning—ICANN 2011*; Springer: Berlin/Heidelberg, Germany, 2011. [CrossRef]
24. Deliège, A.; Cioppa, A.; Van Droogenbroeck, M. HitNet: A neural network with capsules embedded in a Hit-or-Miss layer, extended with hybrid data augmentation and ghost capsules. *arXiv* **2018**, arXiv:1806.06519.
25. Rajasegaran, J.; Jayasundara, V.; Jayasekara, S.; Jayasekara, H.; Seneviratne, S.; Rodrigo, R. DeepCaps: Going Deeper With Capsule Networks. In Proceedings of the 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognitionar, Long Beach, CA, USA, 16–20 June 2019. [CrossRef]
26. Gao, S.H.; Cheng, M.M.; Zhao, K.; Zhang, X.Y.; Yang, M.H.; Torr, P. Res2Net: A New Multi-Scale Backbone Architecture. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 652–662. [CrossRef] [PubMed]
27. Xi, E.; Bing, S.; Jin, Y. Capsule Network Performance on Complex Data. *arXiv* **2017**, arXiv:abs/1712.03480.
28. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**, arXiv:1505.04597.
29. Milletari, F.; Navab, N.; Ahmadi, S. V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation. *arXiv* **2016**, arXiv:1606.04797.
30. Rundo, L.; Han, C.; Nagano, Y.; Zhang, J.; Hataya, R.; Militello, C.; Tangherloni, A.; Nobile, M.S.; Ferretti, C.; Besozzi, D.; et al. USE-Net: Incorporating Squeeze-and-Excitation blocks into U-Net for prostate zonal segmentation of multi-institutional MRI datasets. *Neurocomputing* **2019**, *365*, 31–43. [CrossRef]
31. Rundo, L.; Han, C.; Zhang, J.; Hataya, R.; Nagano, Y.; Militello, C.; Ferretti, C.; Nobile, M.S.; Tangherloni, A.; Gilardi, M.C.; et al. *CNN-Based Prostate Zonal Segmentation on T2-Weighted MR Images: A Cross-Dataset Study*; Springer: Singapore, 2020.
32. Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *39*, 2481–2495. [CrossRef]
33. Woo, S.; Park, J.; Lee, J.-Y.; Kweon, I.S. CBAM: Convolutional Block Attention Module. In Proceedings of the European Conference on Computer Vision (ECCV), Munich, Germany, 8–14 September 2018. [CrossRef]
34. Kather, J.N.; Krisam, J.; Charoentong, P.; Luedde, T.; Herpel, E.; Weis, C.-A.; Gaiser, T.; Marx, A.; Valous, N.A.; Ferber, D.; et al. Predicting survival from colorectal cancer histology slides using deep learning: A retrospective multicenter study. *PLoS Med.* **2019**, *16*, e1002730. [CrossRef]
35. Tsch, L.P.; Rosendahl, C.; Kittler, H. The HAM10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Sci. Data* **2018**, *5*, 180161. [CrossRef]
36. Kermany, D.S.; Goldbaum, M.; Cai, W.; Valentim, C.C.S.; Liang, H.; Baxter, S.L.; McKeown, A.; Yang, G.; Wu, X.; Yan, F.; et al. Identifying Medical Diagnoses and Treatable Diseases by Image-Based Deep Learning. *Cell* **2018**, *172*, 1122–1131.e9. [CrossRef] [PubMed]
37. Bilic, P.; Christ, P.F.; Vorontsov, E.; Chlebus, G.; Chen, H.; Dou, Q.; Fu, C.-W.; Han, X.; Heng, P.-A.; Hesser, J.; et al. The Liver Tumor Segmentation Benchmark (LiTS). *arXiv* **2019**, arXiv:abs/1901.04056.
38. Kingma, D.P.; Ba, J. December. Adam: A Method for Stochastic Optimization, CoRR. *arXiv* **2014**, arXiv:1412.6980.