applied sciences

MDPI

Article

An Accurate Deep Learning-Based Computer-Aided Diagnosis System for Gastrointestinal Disease Detection Using Wireless Capsule Endoscopy Image Analysis

Sameh Abd El-Ghany *, Mahmood A. Mahmood and A. A. Abd El-Aziz

Department of Information Systems, College of Computer and Information Sciences, Jouf University, Sakakah 72388, Saudi Arabia; mamahmood@ju.edu.sa (M.A.M.); aaeldamarany@ju.edu.sa (A.A.A.E.-A.)
* Correspondence: saabdelwahab@ju.edu.sa

Abstract: Peptic ulcers and stomach cancer are common conditions that impact the gastrointestinal (GI) system. Wireless capsule endoscopy (WCE) has emerged as a widely used, noninvasive technique for diagnosing these issues, providing valuable insights through the detailed imaging of the GI tract. Therefore, an early and accurate diagnosis of GI diseases is crucial for effective treatment. This paper introduces the Intelligent Learning Rate Controller (ILRC) mechanism that optimizes the training of deep learning (DL) models by adaptively adjusting the learning rate (LR) based on training progress. This helps improve convergence speed and reduce the risk of overfitting. The ILRC was applied to four DL models: EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2. These models were further enhanced using transfer learning, freezing layers, fine-tuning techniques, residual learning, and modern regularization methods. The models were evaluated on two datasets, the Kvasir-Capsule and KVASIR v2 datasets, which contain WCE images. The results demonstrated that the models, particularly when using ILRC, outperformed existing state-of-the-art methods in accuracy. On the Kvasir-Capsule dataset, the models achieved accuracies of up to 99.906%, and on the Kvasir-v2 dataset, they achieved up to 98.062%. This combination of techniques offers a robust solution for automating the detection of GI abnormalities in WCE images, significantly enhancing diagnostic efficiency and accuracy in clinical settings.

Keywords: gastrointestinal; wireless capsule endoscopy; EfficientNet-B0; balancing; data augmentation

Citation: El-Ghany, S.A.; Mahmood, M.A.; Abd El-Aziz, A.A. An Accurate Deep Learning-Based Computer-Aided Diagnosis System for Gastrointestinal Disease Detection Using Wireless Capsule Endoscopy Image Analysis. *Appl. Sci.* **2024**, *14*, 10243. https://doi.org/10.3390/app142210243

Academic Editors: Jeong Seop Sim and SooJun Park

Received: 26 September 2024
Revised: 24 October 2024
Accepted: 4 November 2024
Published: 7 November 2024

1. Introduction

The small bowel is the mid-part of the GI tract between the stomach and the large bowel. It has a surface area of approximately 30 m^2, including the villi's surface, and is three to four meters long. It is an essential part of the digestive system for absorbing nutrients. As a result, small bowel disorders may result in nutritional deficiencies in children and adults and severe growth retardation in children [1].

The most common types of stomach disorders, including bleeding, pylorus erosion, ulcers, and polyps, require extensive medical attention because stomach abnormalities cause several diseases. In 2018, stomach cancer was one of the top five most common types of cancer worldwide, according to a WHO report [2]. An internal view of the digestive tract is provided by traditional endoscopy. Additionally, the traditional method cannot observe the small intestine due to its complexity and length. Additionally, this endoscopy procedure is uncomfortable and painful for patients.

Another innovative GI diagnostic tool is WCE, an imaging device that captures video frames from the digestive system. It is a tool that does not hurt and has a lot of advantages over other methods, like imaging the small intestine, which is not possible with other traditional endoscopy methods [3]. In addition, WCE provides more realistic images of the digestive system than noninvasive technology like a CT scan [4]. The average number of frames in a video capture is around 8000. WCE uses a camera and transmitter in a small

capsule. In WCE, the capsule device contains a transmitter that detects infections of the gastrointestinal tract.

A capsule can move passively within a patient's gastrointestinal environment or selectively under external control to move and examine the patient's lesions during the examination phase by peristalsis or magnetic fields [5,6]. The WCE device moves through the gastrointestinal tract while transmitting color images at a rate of 2–4 frames per second to a data-recording device [6,7]. Doctors then examine these images to make a diagnosis.

The body's digestive system includes the GI tract, liver, pancreas, and gallbladder, all of which aid in food digestion. The body needs digestion to convert food into nutrients for energy, growth, and cell repair. However, gastrointestinal illnesses pose a significant threat to human health. For instance, gastric cancer, the fourth most prevalent type of cancer worldwide, is the second most common cause of cancer-related mortality worldwide, accounting for 35% of cancer-related deaths [8].

WCE images may show various lesions that are signs of various diseases. Ulcers, lymphoid hyperplasia (LH), bleeding, polyps, angiodysplasia (AD), erythema, and erosion are the most significant lesions [9]. Anatomical landmarks, pathological abnormalities, and poly removal, all important GI disorders, can be seen in WCE images. A more practical method for diagnosing tumors and gastrointestinal hemorrhages, particularly in the small intestine, which is now examined with greater precision [10], is provided by providing a variety of images.

It took a significant amount of time to analyze each patient's captured image [11]. Physicians may overlook these lesions during an examination because they typically only appear in a few frames, and they are small relative to the frame size [12]. In addition, searching through a thousand frames for pathological lesions is a tedious and time-consuming task for physicians [13]. Additionally, there can be a high similarity rate between various contextual images at times; as a result, even the most experienced doctors encounter obstacles that necessitate extensive data analysis.

Despite many of the images being filled with meaningless information, doctors must watch the entire film in order. Because of doctors' carelessness or incompetence, this frequently results in incorrect diagnoses [14,15].

Disorders of the GI system, particularly those of the abdominal, small, and large intestines, affect many people worldwide. In Bangladesh, gastrointestinal infections account for more than 27% of deaths. Inflammatory bowel disease (IBD) affects approximately 1.6 million people in the United States, with a rise of over 200,000 since 2011. Each year, more than 70,000 new IBD cases are discovered. Eighty thousand children have Crohn's disease (CD) or ulcerative colitis (UC). IBD is on the rise worldwide. Cases must be discovered as soon as possible to find a cure for IG disorders. As a result, a computer-aided method (CAD) is required to identify them automatically [16].

Processing images to extract, analyze, and comprehend useful information from a single image or image sequence is the focus of the science known as computer vision (CV). For the machine to see, it aims to develop an artificial system with the capabilities of a human visual system. CV examines images for scenes, objects, faces, and other content in videos, photos, and pictures, using a variety of machine learning (ML) or DL algorithms.

In the last ten years, DL has developed and expanded significantly in tandem with the rapid advancement of technology for automation and visual analysis. Numerous fields have utilized DL methods extensively. Automatic WCE diagnosis and analysis are increasingly using DL algorithms, which have produced excellent outcomes. DL algorithms have the potential to improve diagnostic accuracy while also assisting in reducing physician workload. In the meantime, some researchers use DL to estimate WCE's depth and motion. In WCE data, DL methodologies have demonstrated great potential and will be of great assistance to future WCE technology and enhancing this technology [7].

In this paper, we introduce the ILRC mechanism designed to optimize the training of DL models. The ILRC adaptively adjusts the LR based on the progress of the training process, which improves the speed of convergence and decreases the likelihood of overfitting.

We tested this mechanism on various DL models, including EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2. Furthermore, we enhanced the model layers in conjunction with traditional transfer learning, freezing layers, fine-tuning techniques, residual learning, and modern regularization methods. As a result, the four models achieved better accuracy than other advanced models while requiring fewer parameters and FLOPs. These models were fine-tuned using the Kvasir-Capsule and KVASIR v2 datasets, which consist of images obtained from WCE.

To tackle the problem of the class imbalance of the Kvasir-Capsule dataset, we implemented an under-sampling technique to ensure an even distribution of images across classes. In addition, we applied a data augmentation technique (over-sampling) to generate additional WCE images from existing ones, effectively increasing the size of the two datasets. This technique involved modifying the values of five attributes of the existing WCE images. Moreover, we scaled the resolution of the dataset images to 336 × 336 pixels and normalized the contrast of the images.

The models—EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2—exhibited outstanding performance compared to the current state of the art, achieving accuracies of 99.906%, 98.765%, 98.312%, 99.344%, and 99.719%, respectively, on the Kvasir-Capsule dataset. Additionally, they achieved accuracies of 97.770%, 97.375%, 98%, 98.0625%, and 98.062%, respectively, on the Kvasir-v2 dataset. The integration of the ILRC, traditional transfer learning, freezing layers, fine-tuning techniques, residual learning, and modern regularization methods in our proposed models offers a robust solution for automating the detection of GI abnormalities in WCE images, thereby improving diagnostic efficiency and accuracy in clinical settings. Below is a summary of this research's contributions:

1. We introduced the ILRC mechanism designed to optimize the training of DL models. The ILRC adaptively adjusts the LR based on the progress of the training process, which improves the speed of convergence and decreases the likelihood of overfitting.
2. The integration of the ILRC, traditional transfer learning, fine-tuning techniques, residual learning, and modern regularization methods for the EfficientNet-B0, ResNet101, ResNet101v2, InceptionV3, and InceptionRestNetV2 models offers a robust solution for automating the detection of GI abnormalities in WCE images, thereby improving diagnostic efficiency and accuracy in clinical settings.
3. We lowered the parameter cost by truncating or compressing layers and partially freezing some layers.
4. Our models with the ILRC successfully diagnosed GI disorders, reducing the time and financial resources required for diagnosis and aiding the medical community in promptly and accurately providing appropriate treatment for GI patients.
5. To tackle the problem of class imbalance, we used an under-sampling method on the Kvasir-Capsule dataset. Furthermore, we applied a data augmentation technique to increase the size of the Kvasir-Capsule and Kvasir-v2 datasets that were used to train our model.

The structure of the remaining sections of this paper is as follows: Section 2: A Literature Review on the GI Diagnosis System; Section 3: A Description of the Model's Materials and Architecture; Section 4: Implementation and Evaluation; and Section 5: Conclusion and Suggestions for Future Work.

2. Literature Review

Identifying GI diseases is a prominent area of research within medical image analysis. Numerous studies approach this issue from various angles. For example, in L. Bai et al. [7], a transformer neural network (TNU) with a spatial pooling configuration was used. The self-attention mechanism of a TUN enabled it to capture long-range information effectively, and the exploration of Vision Transformer's (ViT's) spatial structure through pooling had the potential to enhance further ViT's performance on a small-scale capsule endoscopy dataset. For capsule endoscopy disease classification, L. Bai et al. trained the model entirely on two publicly available datasets and achieved good generalization effects with 79.15% accuracy

on the multi-classification task of the Kvasir-Capsule dataset and 98.63% accuracy on the binary classification task of the Red Lesion Endoscopy dataset.

In V. Kumar et al. [15], a hybrid CNN was developed to detect abnormalities by extracting a comprehensive set of meaningful features from wireless capsule endoscopy images using various convolution operations. The architecture consisted of three parallel CNNs, each with distinct feature learning capabilities. The first network employed depthwise separable convolution, while the second utilized cosine normalized convolution. The third network introduced an innovative meta-feature extraction mechanism to identify patterns from the statistical data derived from the features produced by the first two networks, as well as its own preceding layer. This combination of networks effectively managed intra-class variance and successfully identifies GI abnormalities. The proposed hybrid CNN model was trained and evaluated on two widely used publicly available datasets. The results indicated that the model outperformed six leading methods, achieving classification accuracies of 97% and 98% on the KID and Kvasir-Capsule datasets, respectively.

In H. Modi et al. [16], a method for dealing with a difficult dataset was developed using a convolutional neural network (CNN). With an accuracy of 97.82%, the proposed method could classify digestive tract abnormalities into 13 categories. The proposed approach's performance was enhanced when the authors utilized the appropriate optimizer and learning rates. The model's accuracy was also improved as a result of the convolution layer's filter modification. A more balanced dataset for training and higher-quality images could improve the model's accuracy.

In Z. Xiao et al. [17], to generate WCE images from existing WCE images, a WCE-DCGAN network was proposed. On SSD, YOLOv5, and YOLOv4, there were various degrees of performance improvement, achieving an average recognition accuracy of 97.25% on SSD with the images generated by this network and the original images as the input. In the meantime, the diversity of the images produced by WCE-DCGAN not only increased the size of the dataset but also gave the model a good generalization effect.

Using WCE images and a deep CNN, S. Mahmood et al. [18] proposed a method that was both reliable and effective for classifying GI tract anomalies. To accomplish this, the authors proposed the GI Disease-Detection Network (GIDD-Net) custom CNN architecture, which was built from the ground up and has relatively few parameters. Its goal was to detect disorders of the gastrointestinal tract with greater accuracy and efficiency at a lower computational cost. Additionally, by displaying class activation patterns in the bowels as a heat map, the proposed model successfully differentiated GI disorders. The authors in [18] used the synthetic over-sampling technique BORDERLINE SMOTE (BL-SMOTE) to evenly distribute the images among the classes to avoid the issue of class imbalance in the Kvasir-Capsule image dataset. The proposed model achieved the following values for evaluation metrics when compared to a variety of metrics: 98.8%, 98.9%, 98.9%, 99.8%, 0.0474, and 98.8% for precision, accuracy, F1-score, AUC, loss, and recall, respectively.

In A. K. Kundu [13], WCE images were analyzed in normalized RGB color space to investigate bleeding because different shades of red are associated with the human perception of bleeding. An effective region of interest (ROI) was first extracted from the WCE image frame using the interplane intensity variation profile in normalized RGB space in the proposed method. The variation in the normalized green plane from the extracted ROI was then displayed using a histogram. The suggested normalized green plane histograms were used to extract features. The K-nearest neighbor (KNN) classifier was used for classification purposes. Furthermore, morphological operations were used to extract bleeding zones from a bleeding image. A tenfold cross-validation scheme used 2300 WCE images from 30 publicly available WCE videos for performance evaluation. The proposed method outperformed the four reported existing methods with an accuracy of 97.86%, a sensitivity of 95.20%, and a specificity of 98.32%.

S. Fan et al. [19] found small intestinal ulcers and erosion in WCE images by proposing a novel computer-aided detection (CAD) approach based on a deep learning framework. The compression and cropping of images were included in the developed method. The

authors trained the AlexNet convolutional neural network on a database of tens of thousands of WCE images to distinguish between the lesion and normal tissue. The detection of ulcers and erosions yielded results with high specificities of 94.79% and 95.98%, a high sensitivity of 96.80%, and a high accuracy of 95.34%, respectively. In both networks, the area under the receiver operating characteristic curve was greater than 0.98.

S. Charfi et al. [20] proposed an approach to ulcer recognition and detection in WCE images. The authors pre-processed the input WCE images in the first step. Then, they suggested combining color and texture saliency maps by thresholding the resulting map and segmenting it. They discovered that CLBP, PHOG, and BoW produced the best results for feature extraction. A new classification method employing conventional classifiers and the HMM model was fed these features. The proposed method was effective at detecting ulcerous images because it significantly improved classification results with an accuracy of 94.8%.

In A. Caroppo et al. [21], a bleeding detection system utilizing deep transfer learning has been proposed. To extract features, three well-known CNN models—InceptionV3, VGG19, and ResNet50—were employed. Feature selection was performed using the minimum redundancy maximum relevance method. Ultimately, supervised ML methods were applied to classify the selected features into two categories: non-bleeding images and bleeding images. The proposed architecture demonstrated exceptional performance, achieving average accuracies of 97.65% and 95.70% on leading datasets, surpassing the performance of single DL architectures. The optimal combination of accuracy and training time was attained by employing mean value pooling as a fusion rule and SVM as the classifier.

H. Gunasekaran et al. [22] introduced an ensemble model that utilized the predictions from three pre-trained models, DenseNet201, InceptionV3, and ResNet50, which achieved accuracies of 94.54%, 88.38%, and 90.58%, respectively. The predictions from these foundational models were merged using two techniques: model averaging and weighted averaging. After evaluating the model performances on the Kvasir v2 dataset, the authors found that the model averaging ensemble achieved an accuracy of 92.96%, while the weighted average ensemble reached an accuracy of 95.00%.

All the studies mentioned above have the following limitations:

- All the research mentioned above utilized a static LR.
- Training on a small dataset may not provide a comprehensive representation of all possible abnormalities in the digestive system. This can limit the model's ability to generalize to new unseen cases.
- Insufficient data can hinder the model's ability to learn robust features, especially for less frequent conditions, potentially reducing the accuracy and reliability of the model.
- Class imbalance can cause the model to be biased towards the majority class, resulting in poor performance in minority classes. This is particularly problematic in medical diagnosis, where the accurate detection of rare conditions is critical.
- Models trained on limited datasets may overfit the training data, reducing their ability to generalize to new or different datasets.

Our research differs from research mentioned above by introducing the ILRC algorithm to improve the training process of DL models. The ILRC algorithm dynamically adjusts the LR based on the training progress, which leads to faster convergence and reduces the risk of overfitting. We assessed this mechanism across various DL architectures, including EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2. Furthermore, we enhanced these models using transfer learning, freezing layers, fine-tuning techniques, residual learning, and regularization strategies. The models were fine-tuned using the Kvasir-Capsule and KVASIR v2 datasets, which contain images from wireless capsule endoscopy (WCE). Moreover, we implemented several data pre-processing techniques, including under-sampling, augmentation, scaling, and normalization, specifically to address class imbalance. This comprehensive approach to pre-processing for class imbalance is crucial in improving model performance and ensuring a balanced dataset. The proposed

model aims to provide real-time analysis to assist clinicians in preventing emergencies, which is a critical advancement in the practical application of WCE technology.

3. Materials and Methods

3.1. Dataset Description

The Open Science Framework (OSF) has the Kvasir-Capsule dataset [1]. It was created in partnership with the Vestre Viken Health Trust to aid research in computer-assisted diagnosis systems for gastrointestinal endoscopy. The Kvasir-Capsule is a WCE dataset with various labeled images that have been made available to the public. It was made in 2020 and brought up to date in 2021 [15]. The entirety of the dataset's data records is summarized in Table 1. The dataset includes 4,741,621 main data records, comprising 47,238 images with labels, 43 corresponding labeled videos (the videos from which the images were extracted), and 74 un-labeled videos (the videos from which the labeled images were not extracted). Further, 4,694,266 un-labeled images from all of the videos can be extracted. Figure 1 depicts examples of images from the Kvasir-Capsule dataset. Due to unnecessary data duplication, the un-labeled images were not extracted or included in the uploaded data. However, they can be easily extracted from the videos.

Table 1. A description of the Kvasir-Capsule dataset.

Class	Image Count
Labeled images	47,238
Labeled videos	43
Un-labeled images	4,694,266
Un-labeled videos	74

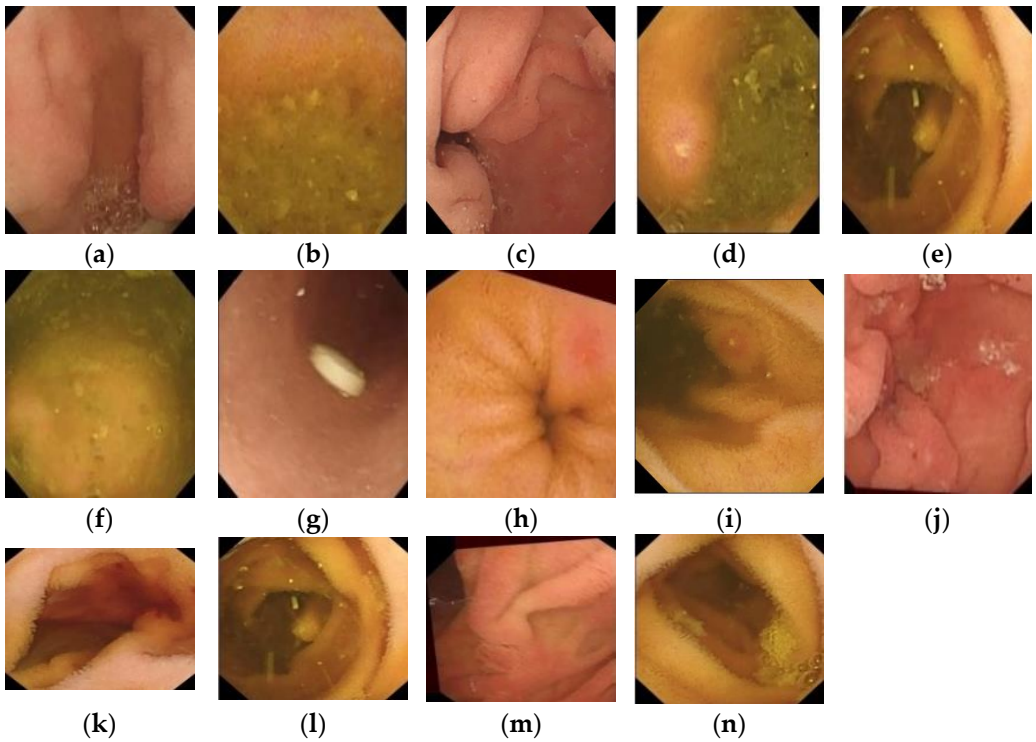

Figure 1. Images from the Kvasir-Capsule dataset: (**a**) polyp; (**b**) reduced mucosal; (**c**) pylorus; (**d**) ulcer; (**e**) lymphangiectasia; (**f**) ileocecal valve; (**g**) foreign body; (**h**) erythema; (**i**) erosion; (**j**) blood hematin; (**k**) blood fresh; (**l**) telangiectasia; (**m**) ampulla of Vater; and (**n**) normal clean mucosa.

Videos were gathered from a series of clinical examinations at the Department of Medicine, the Barium Hospital, and the Vestre Viken Hospital Trust in Norway, which

serves 490,000 people and covers about 200,000 of them. At a resolution of 336 × 336, the videos were initially recorded at a rate of 2 frames per second. The videos were exported in AVI format using the Olympus system's export tool. It was packed, but the export tool changed the specification for the frame rate to 30 fps [18].

The fourteen distinct classes in this dataset were created by extracting video frames from WCE videos. The fourteen distinct classes are ampulla of Vater (class 0), telangiectasia (class 1), blood fresh (class 2), blood hematin (class 3), erosion (class 4), erythema (class 5), foreign body (class 6), ileocecal valve (class 7), lymphangiectasia (class 8), normal mucosa (class 9), polyp (class 10), pylorus (class 11), reduced mucosal (class 12), and ulcer (class 13) [18]. The samples of the Kvasir-Capsule dataset are individual RGB images with three channels and a dimension of 128 × 128 pixels that belong to four distinct classes [23]. Table 1 shows a description of the Kvasir-Capsule dataset.

The Kvasir-Capsule dataset is imbalanced because the normal class clean mucosa contains too many images (34,338) in comparison to other class samples. The class ulcer has 854 images; the class reduced mucosal has 2906 images; the class pylorus has 1529; the class polyp has 55 images; the class lymphangiectasia has 592 images; the class ileocecal valve has 4189 images; the class foreign body has 776 images; the class erythema has 159 images; the class erosion has 506 images; the class blood hematin has 12 images; the class blood fresh has 446 images; the class telangiectasia has 866 images; and the class ampulla of Vater has ten images, as shown in Figure 1 [18].

We divided the Kvasir-Capsule dataset into three sets: a training set with 26,453 images (70%), a test set with 5669 images (15%), and a validation set with 5668 images (15%). Table 2 presents the number of images for each class in the three sets.

Table 2. The distribution of classes in the Kvasir-Capsule dataset.

Class#Id	Class Name	Training Set	Test Set	Validation Set
0	ulcer	478	103	102
1	reduced mucosal	1627	349	349
2	pylorus	856	184	183
3	polyp	31	6	7
4	lymphangiectasia	331	71	71
5	ileocecal valve	2346	503	502
6	foreign body	435	93	93
7	erythema	89	19	19
8	erosion	283	61	61
9	blood hematin	7	1	2
10	blood fresh	250	53	54
11	telangiectasia	485	104	104
12	ampulla of Vater	6	1	1
13	normal clean mucosa	19,229	4121	4120
	Total	26,453	5669	5668

The second dataset is the KVASIR v2 dataset [24]. The Kvasir v2 dataset comprises a collection of images that were annotated and verified by medical professionals, specifically experienced endoscopists. It contains 8000 images classified as eight different classes, namely, esophagitis, dyed lifted polyps, dyed resection margins, normal pylorus, normal z-line, normal, polyps, and ulcerative colitis, as shown in Figure 2. The dataset contains endoscopic images of the GI tract. The samples in each category are evenly distributed, containing 1000 images for each class. Table 3 shows the distribution of Kvasir v2 classes in the test set.

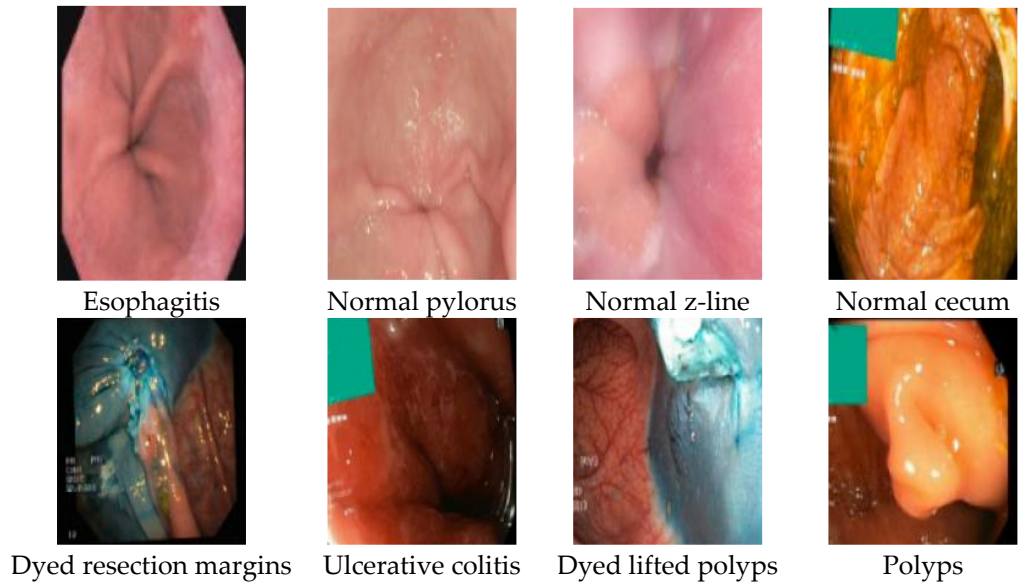

Figure 2. KVASIR v2 dataset.

Table 3. The distribution of classes in Kvasir v2 in the test set.

Class#Id	Class Name	Test Set
0	dyed lifted polyps	154.0
1	dyed resection-margins	134.0
2	esophagitis	140.0
3	normal cecum	141.0
4	normal pylorus	166.0
5	normal z-line	154.0
6	polyps	150.0
7	ulcerative colitis	161.0

3.2. Model Architecture and Training

To identify GI diseases from WCE images, we present the ILRC mechanism aimed at optimizing the training of DL models. The ILRC adjusts the LR adaptively based on the progress of training, which improves convergence speed and reduces the risk of overfitting. The mechanism was tested on several DL models, including EfficientNet-B0, ResNet101, ResNet101v2, InceptionV3, and InceptionResNetV2. Furthermore, we enhanced these models by using traditional transfer learning, fine-tuning techniques, residual learning, and modern regularization methods. The models were fine-tuned with the Kvasir-Capsule and KVASIR v2 datasets for multi-class classification. The architecture of the DL models is illustrated in Figure 3, and the algorithm for fine-tuning the four DL models is described in Algorithm 1, and the algorithm of the ILRC is depicted in Algorithm 2. The steps for implementing the ILRC on the four DL models are as follows: **Phase 1** (pre-processing the Kvasir-Capsule and Kvasir v2 datasets): In the first phase, the Kvasir-Capsule dataset and Kvasir v2 were downloaded and pre-processed. The pre-processing step is critical because it can chip away at the accuracy of the results. In the pre-processing phase, the images of the Kvasir-Capsule dataset were under-sampled, augmented, normalized, and re-scaled. **Phase 2** (splitting the Kvasir-Capsule and Kvasir v2 datasets): In the second phase, the Kvasir-Capsule dataset was divided into three sets: a training set, a test set, and a validation set. The training set consists of 70% of the total data (26,453 samples), the test set consists of 15% (5669 samples), and the validation set consists of 15% (5669 samples). **Phase 3** (pre-training for the four DL models): The third phase was the supervised pre-training step of the transfer learning process. In the supervised pre-training step, the first step of the transfer learning process, the EfficientNet-B0, ResNet101v2, InceptionV3, and

InceptionRestNetV2 models were trained on the ImageNet dataset. During the transfer learning process, we applied a freezing layer. This means that certain layers of a pre-trained model are locked, and their weights remain unchanged during the training on a new task or dataset, such as ImageNet. This is important because it allows us to leverage the general features learned by the model from ImageNet while focusing the training effort on specific parts of the model that are more relevant to the Kvasir-Capsule and Kvasir v2 datasets. **Phase 4** (fine-tuning): In the fine-tuning step, the second step of the transfer learning process, the four DL models were fine-tuned on the training set of the Kvasir-Capsule and Kvasir v2 datasets. **Phase 5** (applying the ILRC): by using the ILRC process, the four DL models were evaluated on the training sets of the Kvasir-Capsule and Kvasir v2 datasets. In this process, we first assessed the training set error. If the training set error was high, we re-trained the model. Conversely, if the training set error was low, we proceeded to calculate the test set error. If the test set error was also high, we re-trained the model again. **Phase 6** (multi-classification using the ILRC): We experimented with multi-classification. Additionally, we evaluated the performance of the four DL models with the ILRC using various metrics including accuracy, specificity, precision, and sensitivity.

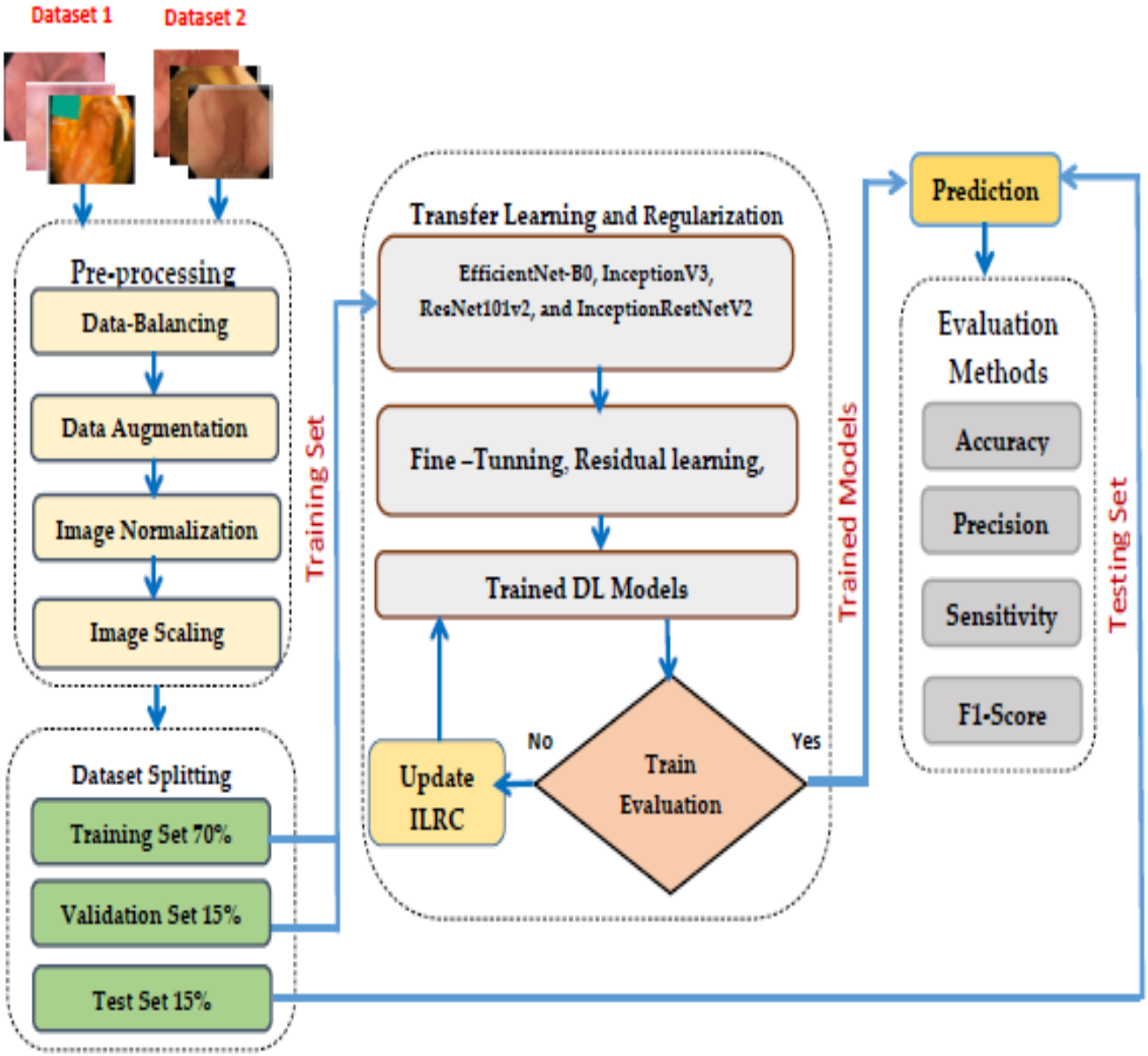

Figure 3. The overall model architecture.

Algorithm 1: The proposed ILRC algorithm for tuning the four DL models.

Input → Kvasir-Capsule or Kvasir v2 dataset *K*
Output ← Fine-tuned four DL models for GI disease classification
BEGIN
  STEP 1: **Pre-Processing of K**
    FOR EACH X **IN** *K***DO**
      Re-scale the dimension of X
      Resize X to 224 × 224 pixels
      Normalize pixel values from the range [0, 255] to [0, 1]
    END FOR
  STEP 2: Data Splitting
    SPLIT *K* **INTO**
      Training set → 70% *of K*
      Testing set → 15% *of K*
      Validation set → 15% *of K*
      IF set's size < 200
        Under-sampling to 200
      ELSE
        IF set's size > 200
          Augmentation to 200
        END IF
      END IF
  STEP 3: Model Pre-Training
    FOR EACH D **IN** [EfficientNet-B0, ResNet101v2, InceptionV3, and Inception ResNetV2] **DO**
      Load and *pre-train* D on the ImageNet dataset
    END FOR
  STEP 4: Model Fine-Tuning
    FOR EACH pre-trained D **DO**
      Fine-tune pre-trained D on the training set
      Utilize the validation set for early stopping, based on the best fine-tuned model performance.
    END FOR
  STEP 5: Applying ILRC
    Apply the ILRC through the training by determining the error of the training set.
    IF the error of the training set is too high,
      GOTO line 26.
    ELSE
      Calculate the error of the test set.
      IF the error of the set is too high,
        GOTO line 26.
      ELSE
        GOTO line 42.
    END IF
    Use the validation set to apply early stopping by assessing the model's best performance.
  STEP 6: Model Assessment
    FOR EACH fine-tuned D **DO**
      Assess D's performance on the test set using measured metrics.
    END FOR
END

3.2.1. Data Pre-Processing

Due to data noise, the final prediction may not be as accurate as the raw data. Pre-processing, which includes under-sampling, data augmentation (over-sampling), scaling, and normalization, is necessary to make the images more suitable for training the proposed models.

Algorithm 2: The algorithm of the ILRC.

```
Input → no_epoch, fr, 1.0 > fr > 0.0
Output ← ILRC
BEGIN
  DO
    DO
      READ N.
      IF (N ≠ 0)
        n_epoch = N
      ELSE
        HALT
      END IF
    WHILE (n_epoch = 0)
  READ well_V
 IF well_V == True THEN
    Compute Cr_valid_loss
    Compute Cr_train_accu
    IF Cr_valid_loss > Pr_valid_loss THEN
      Cr_W = Pr_W
       Cr_B = Pr_B
        N_ILRC = C_ILRC × fr
    END IF
    IF Cr_train_ac < Pr_train_ac THEN
      Cr_W = Pr_W
       Cr_B = Pr_B
        N_ILRC = C_ILRC × fr
    END IF
 END IF
 n_epoch = n_epoch − 1
 WHILE (n_epoch = 0)
END
```

Under-Sampling

Under-sampling refers to a set of methods that aim to even out a classification dataset's class distribution when it is skewed. By keeping all of the data in the minority class and reducing the size of the majority class, under-sampling attempts to balance uneven datasets. It is one of the methods used to obtain more accurate information from datasets that were initially imbalanced. The simplest method for under-sampling is random under-sampling, which removes examples from the training dataset at random from the majority class. This procedure can be repeated until each class has the same number of examples.

Since the images of various diseases in the Kvasir-Capsule dataset were not evenly distributed, and although there were numerous images of common diseases, only a small number of images of rare diseases were utilized for training; we applied the under-sampling technique to solve this issue by decreasing the size of all training, test, and validation sets' classes that had more than 200 images to 200 images.

Data Augmentation

Data augmentation refers to a set of methods for artificially increasing the number of data by creating new data points from existing data or creating new synthetic data from existing data. By creating new and distinct examples to train small datasets, data augmentation can help DL models perform better and produce better results. A DL model performs better and is more accurate when the dataset it uses is extensive and sufficient. Data collection and labeling can be time-consuming and expensive processes for DL models, but the use of data augmentation techniques can reduce these operational costs by transforming datasets.

Simple alterations to visual data are popular for data augmentation. For data augmentation, typical image processing tasks include padding, re-scaling, vertical and horizontal flipping, zooming, random erasing, random rotating, translation (where the image is moved in the X and Y directions), color modification, cropping, gray scaling, changing contrast, and adding noise. Data augmentation improves the model's accuracy, lowering the costs of data collection and labeling, enabling the prediction of rare events, and preventing issues with data privacy.

Since the size of the Kvasir-Capsule and Kvasir v2 datasets was small, we generated WCE images from existing WCE images to increase the dataset's size. We altered the values of five attributes of the existing WCE images using the augmentation method. Table 4 shows the used attributes and their values.

Table 4. The used attributes in the data augmentation techniques and their values.

Attribute	Value
horizontal_flip	True
rotation_range	20
width_shift_range	0.2
height_shift_range	0.2
zoom_range	0.2

By using the data augmentation technique, we increased the number of images in all training set classes of the Kvasir-Capsule dataset that had fewer than 200 images to a total of 200 images each. Specifically, the class "erythema" gained 111 images; the class "blood hematin" increased by 193 images; the class "ampulla of Vater" grew by 194 images; and the class "polyp" had 169 images added. Additionally, the overall size of the Kvasir v2 dataset was expanded from 8000 to 12,000 images.

As a result of applying both under-sampling and data augmentation techniques to the Kvasir-Capsule dataset, each class in the training set was adjusted to contain 200 images. Consequently, the total size of the training set decreased from 26,453 images to 2800 images. Furthermore, the size of the test set was reduced from 5669 images to 2133 images, and the validation set size was also decreased from 5668 images to 2133 images.

Normalization

Since size, pixel-level noise, symbols, bright text, and other characteristics of the images vary, we normalized the images of the Kvasir-Capsule dataset. By normalizing the data in a variety of ways, the effect of various pixel intensities can be defined. From the pixel intensity that is recorded as P_I, the normalized datum P_I* is obtained using the normalization method. For each of the three primary colors, the value of this pixel ranges from 0 to 255. To achieve normalization, the value of each pixel was divided by 255. The experiment's maximum and minimum values served as the foundation for normalization, as shown in Equation (1).

$$P_I^* = (P_{Il} - Min_{old}) \frac{Max_{new} - Min_{new}}{Max_{old} - Min_{old}} + Min_{new},\ l \in [0, n] \quad (1)$$

Scaling

In CV, resizing an image is an important step before processing it. Primarily, small images allow DL models to train more quickly. The neural network must learn from four times as many pixels in a larger input image, which increases the architecture's training time. Therefore, images are frequently resized to a smaller dimension when training DL models so that mini-batch learning can continue, and the computed limitations can be maintained [25]. Common DL models like DenseNet-121, ResNet-50, MobileNetV2, and EfficientNets, which were pre-trained on ImageNet, consistently benefit from this concept. In this research, we scaled the images of the Kvasir-Capsule dataset, as shown in Table 5.

Table 5. The properties of the images of the Kvasir-Capsule dataset after scaling.

Attribute	Value
height	336
width	336
channels	3
batch_size	30

Hence, the images were resized to a fixed 224-by-224 resolution.

3.2.2. Freezing Layers

Freezing layers in DL models is a technique where the weights of specific layers in a pre-trained model are locked, meaning they do not change during further training. This method is important in transfer learning, where a model that has been pre-trained on ImageNet is adapted to a new, usually smaller, dataset. The significance of freezing layers is that they help preserve the general features the model has already learned, such as detecting edges in images, which are often useful across different tasks. By freezing these layers, the model keeps its foundational knowledge, which reduces the risk of overfitting while allowing the other layers to focus on the new task. This technique not only speeds up the training process but also enhances the model's ability to generalize new data [26].

During the transfer learning process, we applied the technique of freezing layers in the four DL models. This approach enabled the remaining layers to focus their training efforts on the Kvasir-Capsule and Kvasir v2 datasets. Additionally, the partial layer freezing method decreased the number of trainable parameters needed by disabling certain layers from re-training. As a result, we maintained the initial features obtained from transfer learning in the upper layers.

3.2.3. Regularization

Regularization is a method used in ML and DL to stop overfitting. Overfitting happens when a model learns to perform well on training data but struggles with new unseen data. Regularization helps by adding extra rules or penalties during training, which prevents the model from fitting too closely to the training examples. This approach allows the model to learn broader patterns, leading to better performance on test data [27].

L1, L2, and dropout are the three main types of regularization [27]: there is L2 regularization (also known as Ridge Regression or Weight Decay), L1 Regularization (or Lasso Regression), and dropout.

L2 regularization adds a penalty to the loss function that is equal to the sum of the squared weights. This helps prevent the model from learning excessively large weights, which can lead to an overly complex model that is likely to overfit training data.

L1 Regularization introduces a penalty equal to the sum of the absolute values of the weights in the loss function. This approach can result in sparse models, where some weights become exactly zero, effectively selecting important features by eliminating those that are not significant.

Dropout is a technique where a portion of the neurons in a layer is randomly "dropped out" or ignored during training. This practice helps to prevent the model from relying too heavily on any single neuron and encourages it to learn more robust and general features. Typically, dropout is applied with a probability parameter (e.g., 0.5), indicating that half of the neurons drop out during training. In our methodology, we used L2 regularization to stop the overfitting for the four DL models.

3.2.4. Network Modification for Transfer Learning

Transfer learning is a method used in DL where a model that has been trained on one task is adapted to work on a different, but often related, task. This approach utilizes the knowledge gained from the first task to improve the learning process and performance in the new task, especially when there is a shortage of labeled data for the second task [28].

In our approach, we fine-tuned the models ResNet101v2, InceptionV3, Inception-ResNetV2, and EfficientNetB0 for accurate GI disease detection. During the tuning phase, we employed the Kvasir-Capsule or Kvasir v2 dataset, splitting the data into training, validation, and testing sets at a ratio of 70:15:15. Each image was labeled accordingly, and we set up the input and output layers for training. This method was applied consistently across all the four DL models.

For the EfficientNetB0 model, we used the input convolution layer (conv_1), which contained 64 filters sized at $3 \times 3 \times 64$. The output layer chosen was FC8. An activation function was applied to this layer, and we trained a modified CNN that concentrated solely on GI image features. We added a batch normalization layer for the base model with momentum and epsilon parameters. After the batch normalization layer, a dropout of 50% was applied to reduce the overfitting of the model. Additionally, the last two layers, specifically the classification and Softmax layers, were modified. Consequently, the length of the output is equal to the number of GI classes.

For the ResNet101v2, InceptionV3, and InceptionResNetV2 models, the last two layers were also removed. The convolutional layer (conv_1) was used as the input layer, featuring 64 filters of size $3 \times 3 \times 64$. We again selected the FC8 layer as the output layer, applying the activation function softmax. This layer experienced activation, resulting in the training of a modified CNN that focused exclusively on the features of GI images.

3.2.5. Resnet101v2

Resnet101 and Resnet101v2 are enhanced convolutional neural networks (CNNs) from the deep residual networks (Resnet) with 101 layers. These architectures were introduced to address the challenges of training very deep neural networks (DNNs), which were prone to the vanishing gradient problem. ResNet introduced the concept of residual learning, allowing for a more effective training of extremely deep networks. They were trained on the ImageNet database. The two models learned rich feature representations for a variety of images as a result. The two models accept images 224×224 pixels in size with three RGB colors. The concept of skip connections, also known as fast-forward connections, was utilized by Resnet. Before applying the rectified linear unit (ReLU) activation, the idea of feeding deeper layers from earlier layers was considered. The risk of vanishing gradient issues in the network is reduced by this shortcut or skip connection. In Resnet, the identity or residual block has both a forward and a fast-forward connection. The fact that ResNet101v2 uses batch normalization before each weight layer is the primary distinction between ResNet101v2 and ResNet101 [29].

The mathematical model for ResNet101 involves the specifications of the individual building blocks and the overall architecture of the network. Resnet is composed of residual blocks, and ResNet101, in particular, has 101 layers. The following is a simplified representation of the mathematical model for a basic residual block in ResNet:

Let x be the input to the block, and $F(x)$ represents the residual function to be learned. The output y of the residual block is given by the following:

$$y = F(x) + x \tag{2}$$

The residual function $F(x)$ is typically implemented as a series of convolutional layers with batch normalization and activation functions. This can be written as follows:

$$F(x) = W_{2\sigma}\left(W_{1x}\right) \tag{3}$$

In ResNet101v2, improvements are often introduced, such as pre-activation, which means that batch normalization and activation are applied before the convolutional layer. This can be written as follows:

$$F(x) = \sigma(W_2.BatchNorm(W_1.x)) \tag{4}$$

where W_1 and W_2 are the learnable weights of the convolutional layers, and σ represents the activation function such as ReLU.

3.2.6. InceptionV3

InceptionV3 is a deep CNN architecture that belongs to the Inception family. It was introduced by Google researchers as an extension and improvement over the original Inception architecture. InceptionV3 is designed for image classification and object recognition tasks and is known for its efficient use of computational resources. InceptionV3 introduces the concept of Inception modules, which are blocks containing multiple parallel convolutional branches with different filter sizes. This allows the network to capture features at different scales. InceptionV3 uses factorization techniques, such as 1×1 convolutions, to reduce the number of parameters in the network and make it computationally efficient. Batch normalization is applied after each convolutional layer, and ReLU activation functions introduce non-linearity.

InceptionV3 includes auxiliary classifiers at intermediate layers during training. These auxiliary classifiers help with the training process by providing additional supervision signals and gradients. Instead of traditional fully connected layers at the end of the network, InceptionV3 uses global average pooling. This reduces the number of parameters and helps make the model more robust to spatial translations and distortions.

The overall architecture of InceptionV3 consists of a series of Inception modules, and the network is trained for image classification tasks using datasets such as ImageNet. It achieved competitive performance with relatively fewer parameters compared to previous architectures [29].

A simplified representation of the mathematical model for a basic Inception module in InceptionV3 is as follows: Let X be the input tensor; $W_{1\times1}$, $W_{3\times3\_reduce}$, $W_{3\times3}$, $W_{5\times5\_reduce}$, $W_{5\times5}$, $W_{pool-proj}$ represent the learnable weights of the convolutional filters; γ and β represent the scale and shift parameters for batch normalization; and σ represents the activation function.

1. The 1×1 Convolution Branch:

$$X_{1\times1} = Conv2D(X,\ W_{1\times1}, padding = \text{'same'},\ activation = \text{'relu'}) \tag{5}$$

2. The 3×3 Convolution Branch:

$$X_{3\times3\_reduce} = Conv2D(X,\ W_{3\times3\_reduce}, padding = \text{'same'},\ activation = \text{'relu'}) \tag{6}$$

$$X_{3\times3} = Conv2D\left(X,\ W_{3\times3_{reduce}}, W_{3\times3}, padding = \text{'same'},\ activation = \text{'relu'}\right) \tag{7}$$

3. The 5×5 Convolution Branch:

$$X_{5\times5\_reduce} = Conv2D(X,\ W_{5\times3\_reduce}, padding = \text{'same'},\ activation = \text{'relu'}) \tag{8}$$

$$X_{5\times5} = Conv2D(X,\ W_{5\times5\_reduce}, W_{5x5}, padding = \text{'same'},\ activation = \text{'relu'}) \tag{9}$$

4. The Max Pooling Branch:

$$X_{pool} = MaxPooling2D(X, pool_{size} = strides\ \ (1,1),\ padding = \text{'same'}) \tag{10}$$

$$X_{pool\_proj} = Conv2D\Big(X_{pool},\ W_{pool}\ ,\ padding = \text{'same'},\ activation = \text{'relu'}\Big) \tag{11}$$

5. The Concatenation of Branch Outputs:

$$X_{out} = Concatenate\Big(\Big[X_{1\times1},\ X_{3\times3}, X_{5\times5},\ X_{pool\_proj}\Big]\Big) \tag{12}$$

The overall InceptionV3 architecture is composed of multiple repeated Inception modules, auxiliary classifiers, global average pooling, and a final fully connected layer for classification.

3.2.7. InceptionResNetV2

The InceptionResNet architecture combines elements of the Inception architecture with residual connections inspired by ResNet. This hybrid architecture aims to leverage the benefits of both Inception's multi-scale feature extraction and ResNet's residual learning, resulting in a powerful and effective deep neural network. The InceptionResNet architecture was introduced to improve training convergence and overall performance. The core building blocks in InceptionResNet are Inception modules, like those in InceptionV3, but with added residual connections. Each Inception module consists of parallel convolutional branches with different filter sizes. The output of each branch is combined with the original input through a residual connection. InceptionResNet employs factorized convolutions, including 1×1 convolutions, to reduce the number of parameters and enhance computational efficiency.

Batch normalization is applied after each convolutional layer to improve training stability. Activation functions, typically ReLU, introduce non-linearity into the model. Residual connections are added to facilitate the flow of gradients during backpropagation, addressing the vanishing gradient problem. The residual connections allow the network to learn residual functions, making it easier to train very deep networks.

Similar to InceptionV3, InceptionResNet uses global average pooling instead of fully connected layers at the end of the network. InceptionResNet typically includes a stem network for initial feature extraction. Reduction blocks are used to reduce spatial dimensions and computational complexity.

The combination of Inception modules and residual connections in InceptionResNet aims to create a more efficient and trainable architecture for image classification and other computer vision tasks [29].

The mathematical model for a simplified version of an InceptionResNet block is similar to the simplified representation of the mathematical model for a basic Inception module in InceptionV3 that was mentioned above, with an additional equation that can be expressed as follows:

Residual Connection:

$$X_{out} = X + X_{out} \tag{13}$$

In this block, each branch applies convolutional operations with different filter sizes, and the results are concatenated. The residual connection allows the model to learn residuals and facilitates the training of deep networks.

3.2.8. EfficientNe-B0

EfficientNet-B0 is part of the EfficientNet family of neural network architectures, which are designed to achieve state-of-the-art accuracy while being computationally efficient in terms of parameters and computation cost. EfficientNet was proposed by Mingxing Tan and Quoc V. Le. It serves as the foundational model within the EfficientNet family, which comprises eight variants denoted as B0 to B7. Each variant differs in terms of dimensions and performance. EfficientNet-B0, for instance, contains 5.3 million parameters and achieves a top-1 accuracy of 77.3% on the ImageNet dataset [30].

EfficientNet introduces a compound scaling method that optimally scales the model's depth, width, and resolution simultaneously. This method allows EfficientNet models to achieve better accuracy than traditional models with a similar number of parameters [31]. The scaling factors are controlled by three parameters: $\varnothing$ for depth, β for width, and γ for resolution. EfficientNet-B0 has $\varnothing = 1$, $\beta = 1$, and $\gamma = 1$.

EfficientNet-B0 builds upon the inverted bottleneck residual blocks of MobileNetV2. Additionally, it incorporates squeeze-and-excitation blocks, which bolster the model's ability to learn expressive features by adaptively recalibrating the channel-wise feature

responses [32,33]. EfficientNet-B0 uses the Swish activation function, which is a smooth, non-monotonic activation function designed to improve model performance. The network ends with a global average pooling layer, followed by a fully connected layer for the final classification. EfficientNet-B0 includes regularization techniques like dropout to prevent overfitting.

A simplified representation of the mathematical model for a basic building block in EfficientNet-B0 is as follows:

1. Depthwise Separable Convolution:

$$X_d = DepthwiseConv2D(X, W_d,\ padding = \text{'same'},\ activation = None) \tag{14}$$

Batch Normalization after Depthwise Convolution:

$$X_b n_d = BatchNormalization(X_d, \gamma_d, \beta_d) \tag{15}$$

2. Swish Activation after Depthwise Batch Normalization:

$$X_s\ wish = \sigma(X_{bn_d}) \tag{16}$$

3. Pointwise Convolution:

$$X_p = Conv2D(X_s wish, W_p,\ adding = \text{'same'},\ activation = None) \tag{17}$$

4. Batch Normalization after Pointwise Convolution:

$$X_b n_p = BatchNormalization(X_p,\ \gamma_p, \beta_p) \tag{18}$$

5. Skip Connection (Identity Connection):

$$X_{out} = X + X_{bn_p} \tag{19}$$

This block, often referred to as a depthwise separable convolution block with Swish activation, is a fundamental component repeated multiple times in the EfficientNet-B0 architecture.

4. Model Implementation and Evaluation

4.1. Model Evaluation Metrics

The proposed models were evaluated using accuracy, precision, recall, and the F1-score. The used measured metrics are defined in Equations (20)–(23).

$$\text{Accuracy} = (\text{TP} + \text{TN})/(\text{TP} + \text{TN} + \text{FP} + \text{FN}) \tag{20}$$

$$\text{Precision} = \text{TP}/(\text{TP} + \text{FP}) \tag{21}$$

$$\text{Recall} = \text{TP}/(\text{TP} + \text{FN}) \tag{22}$$

$$\text{F1-score} = 2 \times (\text{Precision} \times \text{Recall})/(\text{Precision} + \text{Recall}) \tag{23}$$

TP, TN, FP, and FN represent true positive, false negative, false positive, and false negative, respectively. The degree to which the result corresponds to the actual value that we were aiming for is called accuracy. The proportion of positive samples correctly predicted by the model is what is meant by the term precision. The ratio of the number of actual positives to the number of true positives is what is meant by the term recall. The harmonic average of recall and precision is known as the F1-score.

4.2. Model Implementation

We conducted three experiments using the Kvasir-Capsule and Kvasir v2 datasets. In the first experiment, we integrated the ILRC process with the EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2 DL models. This integration allows for the adaptive adjustment of the LR based on the training progress, which enhances convergence speed

and reduces the risk of overfitting. We again evaluated the four DL models on the Kvasir v2 dataset after applying data augmentation techniques.

In the second and third experiments, we continued to combine the ILRC process with the four DL models. In the second experiment, we did not use under-sampling or data augmentation techniques on the Kvasir-Capsule dataset. However, in the third experiment, we applied both under-sampling and data augmentation techniques on the same dataset.

For our experiments, the Kvasir-Capsule and Kvasir datasets were split into 70% for training, 15% for testing, and 15% for validation. All three experiments were performed in the Kaggle environment. Table 6 lists the specifications of the computer used in our experiments.

Table 6. The used PC's description.

Item	Description
CPU	Intel(R) Xeon(R) CPU @ 2.20 GHz
GPU	NVIDIA Tesla P100 with 16 GB of memory
RAM	32 GB of memory
OS	Ubuntu 2024 x86_64 GNU/Linux

In the first experiment, we applied the data augmentation technique to the balanced Kvasir v2 dataset, and we integrated the ILRC process with the EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2 DL models. This integration allows for the adaptive adjustment of the LR based on the training progress, which enhances convergence speed and reduces the risk of overfitting. The Kvasir v2 dataset contains 8000 images classified as eight different classes, namely, esophagitis, dyed lifted polyps, dyed resection margins, normal pylorus, normal z-line, normal, polyps, and ulcerative colitis.

Table 7 shows the results of the first experiment conducted with EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2. For multi-classification, we used the four CNN models on the Kvasir v2 dataset to distinguish images from eight classes.

Table 7. The results of multi-classification for the balanced Kvasir v2 dataset.

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
EfficientNet-B0	97.770	91.071	91.107	91.061
ResNet101V2	97.375	89.506	89.623	89.471
InceptionV3	98	92.004	92.071	92
InceptionResNetV2	98.062	92.284	92.290	92.242

According to Table 7, the average accuracies for the models EfficientNet-B0, ResNet101, ResNet101V2, InceptionV3, and InceptionResNetV2 were 97.770%, 97.375%, 98%, and 98.062%, respectively. Among these models, InceptionResNetV2 achieved the highest averages for accuracy, recall, precision, and the F1-score, with values of 98.062%, 92.284%, 92.290%, and 92.242%, respectively. On the other hand, the ResNet101V2 model had the lowest averages for these metrics, recording 97.375% for accuracy, 89.506% for recall, 89.623% for precision, and 89.471% for the F1-score.

Figure 4 shows the training and validation loss for each epoch of both the training and validation sets across four DL models that utilize the ILRC algorithm. For EfficientNet-B0, between 0 and 2.5 epochs, both the training and validation losses begin high and decrease quickly, which is typical as the model starts to learn patterns from the data. From 2.5 to 7.5 epochs, the validation loss experiences some fluctuations with noticeable spikes, while the training loss continues to decrease steadily. These spikes in the validation loss indicate that the model may be struggling to generalize to unseen data, a common occurrence in the early phases of training. This suggests that the model might be starting to overfit the training data slightly. However, these fluctuations stabilize as training progresses. From 7.5

to 17.5 epochs, both the training and validation losses become more stable, with the training loss continuing to decrease slightly and the validation loss leveling off. The validation loss remains slightly higher than the training loss, suggesting that while the model has learned effectively, it may be beginning to overfit the training data. The small gap between the two losses indicates that overfitting is not significant. The stability of the validation loss during this phase is encouraging, showing that the model has reached a point of reasonable generalization. Overall, the model displays positive learning behavior, as evidenced by the consistent decrease in both training and validation losses.

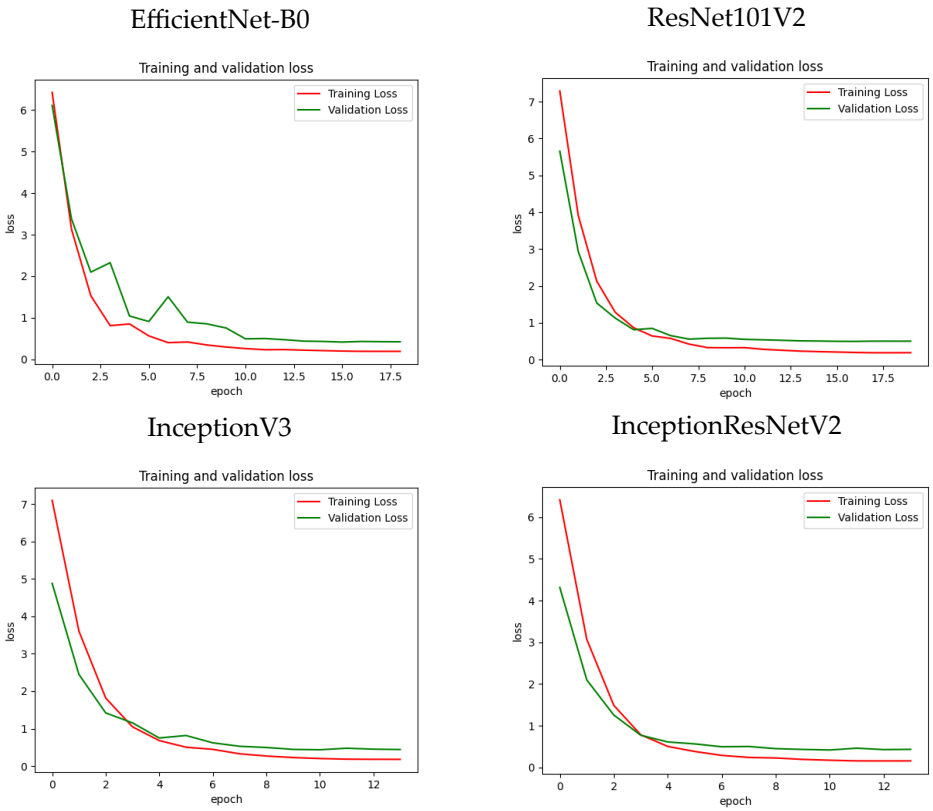

Figure 4. The training and validation loss of the four CNN models for each epoch using the test set with the ILRC.

For the models ResNet101V2, InceptionV3, and InceptionResNetV2, both the training and validation loss begin at a high level and decrease quickly during the first few epochs. This pattern is common as the models start to recognize the patterns in the training data. Throughout this phase, the training and validation losses are closely aligned, indicating effective learning without overfitting at this stage. As training continues, both losses keep decreasing, although the rate of decline begins to slow down.

Around epoch 5, there is a noticeable fluctuation in the validation loss, which may suggest that the models are starting to experience some instability when trying to generalize to new unseen data. This is a typical occurrence during training, especially as models refine their understanding of data. Eventually, both the training and validation losses level off, with training loss showing a gradual decrease while validation loss stabilizes. The small gap between the training and validation losses indicates that the model is not significantly overfitting. However, there is a slight increase in the validation loss toward the end, which could signal an early indication of overfitting if training were to continue for additional epochs. Overall, the models demonstrate good learning behavior, with both training and validation losses decreasing steadily and remaining closely aligned for the majority of the training process.

Figure 5 displays the training and validation accuracy for the four models: EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2. For EfficientNet-B0, from 0 to 2.5 epochs, both the training and validation accuracy start at low levels and increase rapidly during the initial epochs. The training accuracy climbs quickly, while the validation accuracy experiences more fluctuations, particularly between epochs 1 and 3. The validation accuracy curve shows significant oscillations, indicating that the model is having difficulty generalizing well during the early training stages. This may be attributed to the model's attempts to adapt to the validation data, resulting in varying performance.

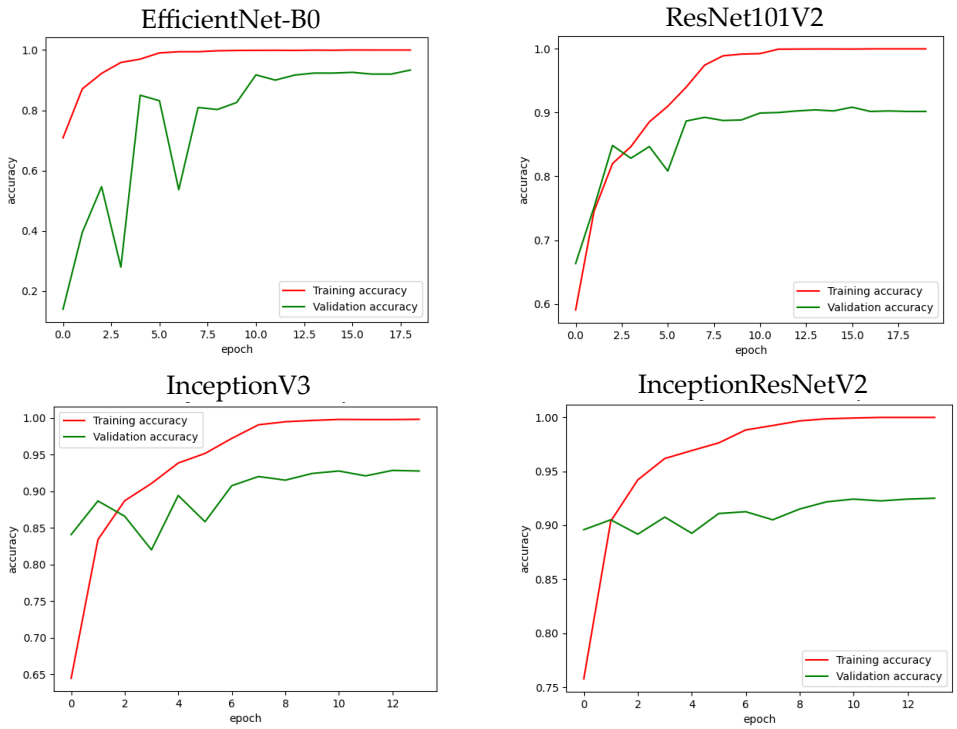

Figure 5. The training and validation accuracy of the four CNN models for each epoch using the test set with the ILRC.

From 2.5 to 7.5 epochs, the training accuracy continues to improve and eventually stabilizes near 1.0, suggesting that the model is fitting the training data very well. However, the validation accuracy continues to fluctuate significantly, with noticeable increases and decreases.

ResNet101V2 performance analysis shows that from 0 to 3 epochs, both the training and validation accuracy start at around 0.6 and increase rapidly. Initially, the training accuracy is slightly lower than the validation accuracy, which is typical as the model begins to adapt to the training data. By about epoch 3, the validation accuracy briefly exceeds the training accuracy, indicating that the model is learning general features that are effective for both datasets.

From 3 to 10 epochs, the training accuracy continues to rise, approaching 1.0, which shows that the model fits the training data very well. The validation accuracy stabilizes around 0.9 but exhibits minor fluctuations, suggesting that the model is encountering difficulties in maintaining consistent performance on the validation dataset while optimizing for the training data.

From 10 to 18 epochs, the training accuracy plateaus near 1.0, indicating that the model has nearly perfectly learned the training data. Meanwhile, the validation accuracy remains steady around 0.9, with slight fluctuations. Overall, the model demonstrates significant learning progress, with both training and validation accuracy improving rapidly during the early epochs.

InceptionV3 training and validation accuracy analysis shows that during the first 0 to 3 epochs, both the training and validation accuracy start off relatively low but increase rapidly. The validation accuracy shows fluctuations during this time, suggesting that the model is refining its understanding of the data. Notably, the validation accuracy is higher than the training accuracy early on, indicating that the model generalizes well at this stage, potentially due to the regularization effect from using validation data.

From 3 to 8 epochs, the training accuracy continues to rise and nearly reaches 1.0, demonstrating that the model is effectively fitting the training data. However, the validation accuracy displays some instability with fluctuations, which may indicate the model's sensitivity to varying batches of validation data.

Between 8 to 13 epochs, the training accuracy stabilizes close to 1.0, indicating that the model has almost perfectly learned the training data. The validation accuracy stabilizes around 0.90 to 0.95, although it still experiences minor fluctuations. Overall, the model initially generalizes well to the validation data, as reflected by the high validation accuracy observed early in the training process.

For InceptionResNetV2, the training accuracy rises quickly in the early epochs, showing that the model learns the training data patterns effectively. After around 5 epochs, the training accuracy levels off near 1.0, indicating a strong fit to the training data. In contrast, the validation accuracy starts to increase but then fluctuates between 0.90 and 0.92 after a few epochs.

Figure 6 depicts the confusion matrices for the four CNN models over the test set of the balanced Kvasir v2 dataset. The test set was divided into eight classes: namely, esophagitis (313 images), dyed lifted polyps (333 images), dyed resection margins (291 images), normal pylorus (269 images), normal z-line (300 images), normal (300 images), polyps (308 images), and ulcerative colitis (286 images). The rows of the confusion matrix represent the actual classes. The columns represent the predicted classes by the model. The diagonal elements (from top-left to bottom-right) indicate the number of correct predictions for each class. Off-diagonal elements indicate misclassifications, where the model predicted the wrong class.

The EfficientNet-B0 model correctly predicted the following number of instances for each class: 137 for class 0, 132 for class 1, 111 for class 2, 138 for class 3, 155 for class 4, 132 for class 5, 141 for class 6, and 147 for class 7. The ResNet101V2 model accurately predicted the number of instances for each class as follows: 141 for class 0, 129 for class 1, 105 for class 2, 138 for class 3, 156 for class 4, 122 for class 5, 134 for class 6, and 149 for class 7.

The InceptionV3 model accurately predicted the number of instances for each class as follows: 139 for class 0, 134 for class 1, 123 for class 2, 132 for class 3, 155 for class 4, 125 for class 5, 145 for class 6, and 151 for class 7. The InceptionResNetV2 model accurately predicted the following number of instances for each class: 142 for class 0, 133 for class 1, 119 for class 2, 139 for class 3, 156 for class 4, 130 for class 5, 140 for class 6, and 148 for class 7.

Table 8 shows the results of the second experiment conducted with EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2. In this experiment, we integrated the ILRC process with the four DL models. The Kvasir-Capsule dataset was utilized to evaluate the performance of the models prior to implementing under-sampling and data augmentation techniques. For multi-classification, we used the four CNN models on the Kvasir-Capsule dataset to distinguish images from fourteen distinct classes named normal mucosa, ampulla of Vater, telangiectasia, blood fresh, blood hematin, erosion, erythema, foreign body, ileocecal valve, lymphangiectasia, polyp, pylorus, reduced mucosal, and ulcer.

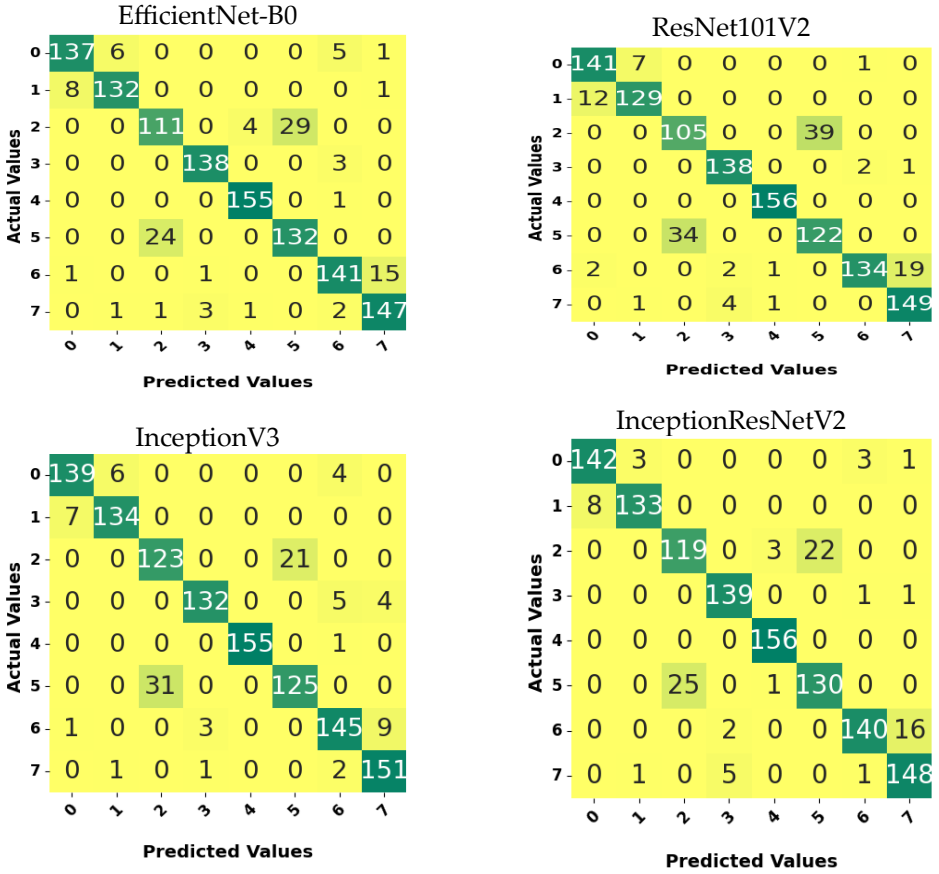

Figure 6. The confusion matrices for the four CNN models using the test set with the ILRC.

Table 8. The results of multi-classification for the imbalanced Kvasir-Capsule dataset using the ILRC.

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
EfficientNet-B0	99.295	99.306	99.295	99.261
ResNet101V2	98.413	98.445	98.413	98.384
InceptionV3	97.354	97.363	97.354	97.091
InceptionResNetV2	98.765	98.628	98.765	98.598

According to Table 8, the accuracy averages of EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2 were, respectively, 99.295%, 98.413%, 97.354%, and 98.765%. Accuracy, recall, precision, and F1-score averages were all the highest in the EfficientNet-B0 model, with 99.295%, 99.306%, 99.295%, and 99.261%, respectively. Accuracy, recall, precision, and F1-score averages were all the lowest in the InceptionV3 model, with 97.354%, 97.363%, 97.354%, and 97.091%, respectively.

Figures 7 and 8 illustrate the loss and accuracy for each epoch of both the validation and training datasets. After approximately epoch 17.5, the training accuracy of the EfficientNet-B0 model neared 100%, while the validation accuracy surpassed 98%. The training and validation losses were comparable, fluctuating between 0.0 and 0.5. For the ResNet101V2 model, both the training and validation accuracies exceeded 95%, with losses ranging from 0.0 to 1. The InceptionV3 model achieved a training accuracy of over 98%, with validation accuracy also at 98%. The training and validation losses for InceptionV3 varied between 0.0 and 1. The InceptionRestNetV2 model recorded training and validation accuracies above 95%, with losses also falling within the range of 0.0 to 1.

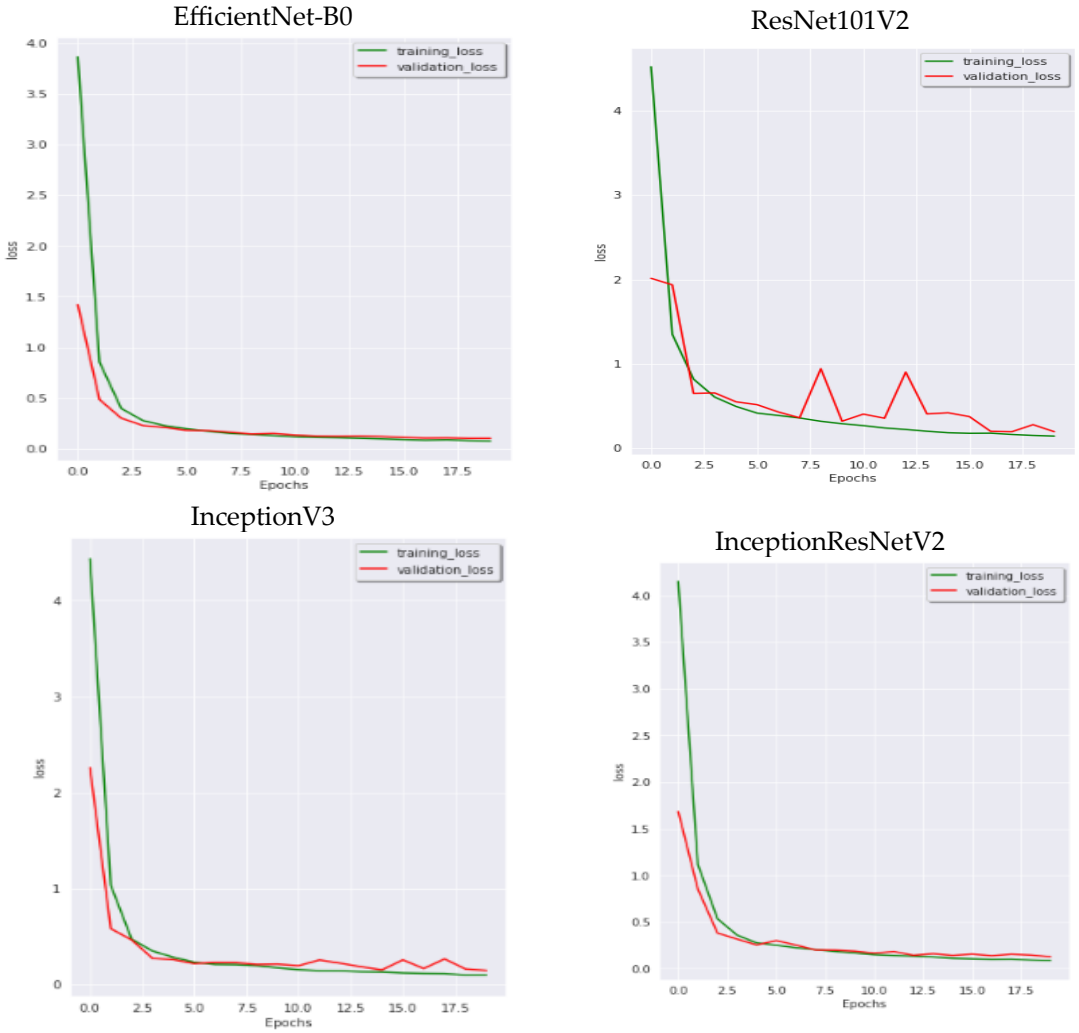

Figure 7. The training and validation loss of the four CNN models for each epoch using the imbalanced Kvasir-Capsule dataset.

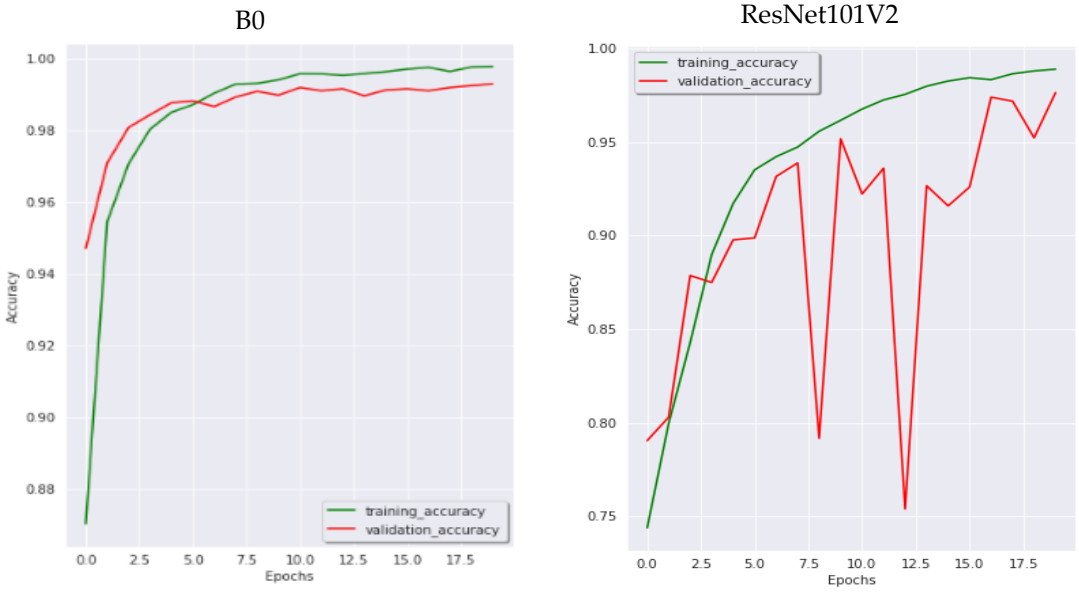

Figure 8. *Cont.*

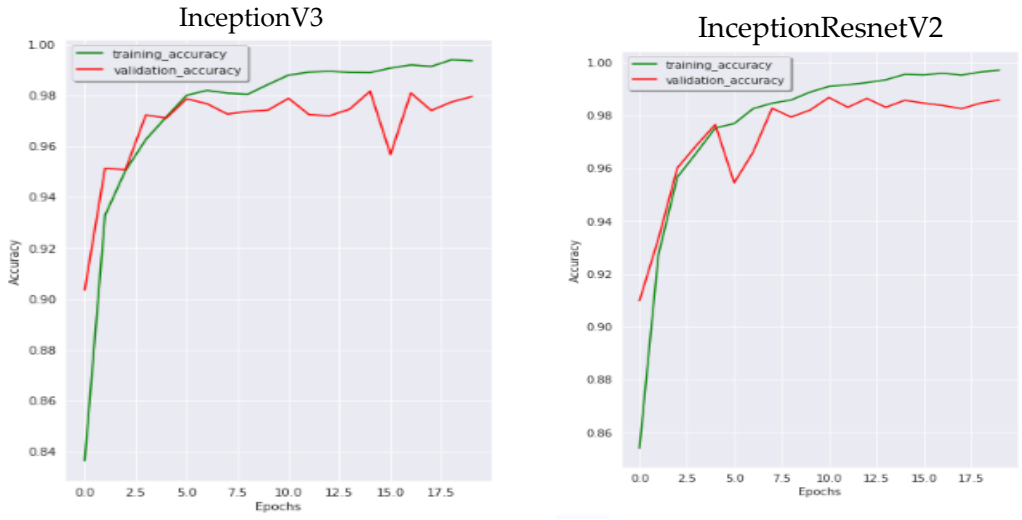

Figure 8. The training and validation accuracy of the four CNN models for each epoch using the imbalanced Kvasir-Capsule dataset.

The confusion matrices for the four CNN models in the imbalanced test set are depicted in Figure 9. The test dataset was divided into fourteen categories: normal mucosa (4121 images), ampulla of Vater (1 image), telangiectasia (104 images), blood fresh (53 images), blood hematin (1 image), erosion (61 images), erythema (19 images), foreign body (93 images), ileocecal valve (503 images), lymphangiectasia (71 images), polyp (6 images), pylorus (184 images), reduced mucosal (349 images), and ulcer (103 images).

With an accuracy of 99.8%, EfficientNet-B0 correctly predicted 4115 images out of 4121 for the class normal mucosa. With an accuracy of 99.2%, the ResNet101V2 model correctly predicted 4092 images. With an accuracy of 99.5%, the InceptionV3 model correctly predicted 4101 images. The InceptionResNetV2 model was 99.8% accurate and correctly predicted 4116 images.

The EfficientNet-B0 model correctly predicted 493 images out of 503, giving it an accuracy of 98% for the class ileocecal valve. The ResNet101V2 and InceptionV3 models correctly predicted 481 images, giving them an accuracy of 95.6%. The InceptionResNetV2 model was 97% accurate in predicting 488 images.

Out of 184 images, the EfficientNet-B0 model correctly predicted 175, achieving an accuracy of 95.1% for the class pylorus. With an accuracy of 84.2%, ResNet101V2 correctly predicted 155 images. With an accuracy of 91.3%, the InceptionV3 model correctly predicted 168 images. A total of 174 images were predicted with 94.5% accuracy by the InceptionResNetV2 model.

After utilizing the imbalanced small Kvasir-Capsule dataset, we conducted a third experiment where we combined the ILRC process with the four DL models. In this experiment, we implemented under-sampling and data augmentation (over-sampling) techniques on the imbalanced small Kvasir-Capsule dataset. In the balanced Kvasir-Capsule dataset, ten classes of the test set have 200 images. The results of the third experiment for the four CNN models with the balanced Kvasir-Capsule dataset, including EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionRestNetV2, are shown in Table 9.

Table 9. The results of multi-classification for the balanced Kvasir-Capsule dataset using the ILRC.

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
EfficientNet-B0	99.906	99.929	99.929	99.929
ResNet101V2	98.765	98.765	98.628	98.598
InceptionV3	98.312	97.835	98.765	98.244
InceptionResNetV2	99.344	99.500	99.509	99.500

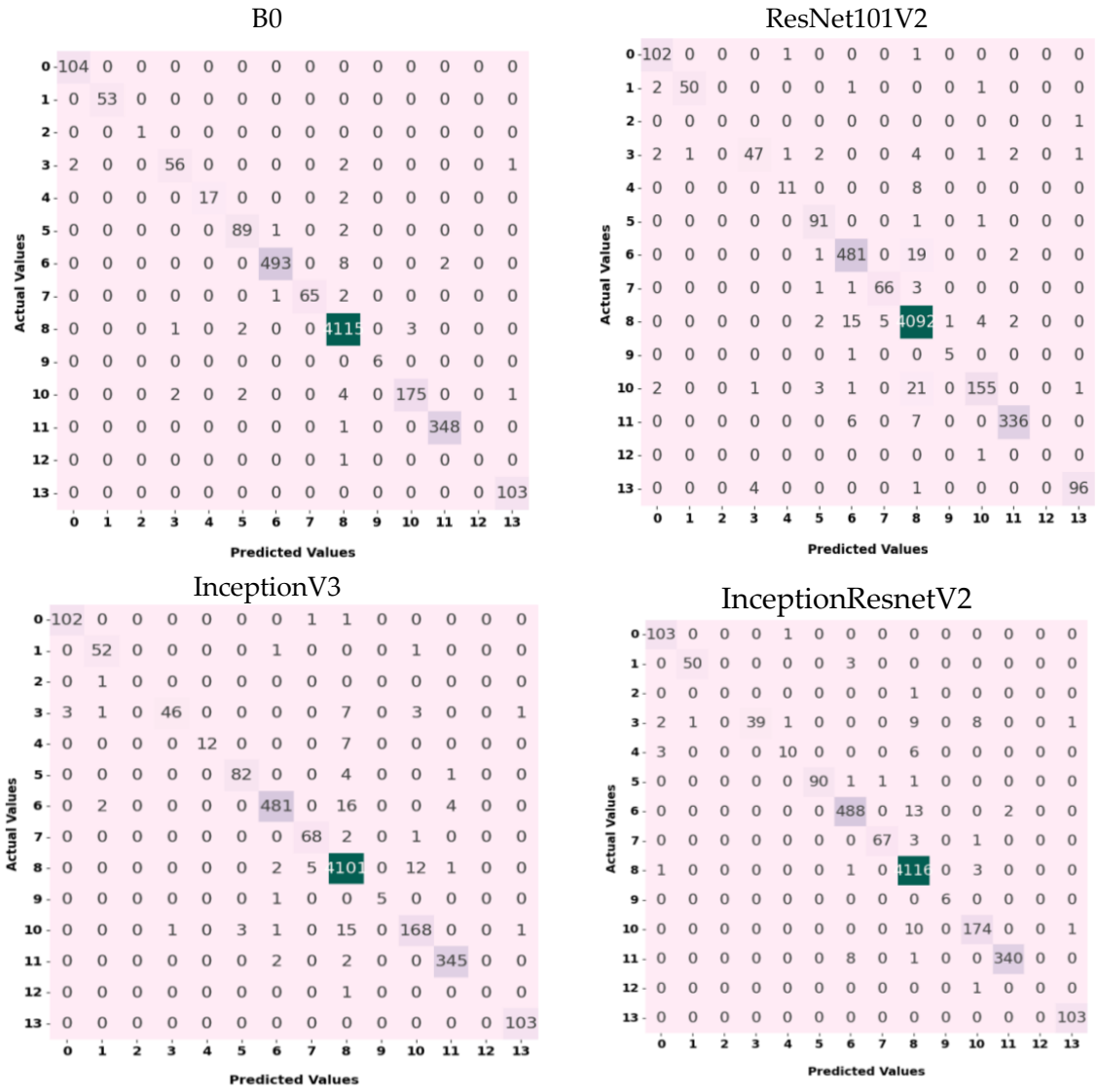

Figure 9. The confusion matrices for the CNN models using the imbalanced Kvasir-Capsule dataset.

As shown in Table 9, the average accuracy rates for EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2 are 99.906%, 98.765%, 98.312%, and 99.344%, respectively. The EfficientNet-B0 model achieved the highest average accuracy, recall, precision, and F1-score, with values of 99.906%, 99.929%, 99.929%, and 99.929%. In contrast, the InceptionV3 model recorded the lowest average accuracy, recall, and F1-score, which are 98.312%, 97.835%, and 98.244%, respectively. Additionally, the ResNet101V2 model had the lowest average precision at 98.628%.

The loss and accuracy for each epoch of the validation and training sets are shown in Figures 10 and 11. The EfficientNet-B0 model's training and validation accuracies were close to 100% after epoch 15. The training and validation loss ranged between 0 and 1. The ResNet101V2 model achieved a training accuracy close to 100% and achieved a validation accuracy ranging between 80 and 100%. ResNet101V2's training and validation loss ranged from 0 to 2. The InceptionV3 model had a training and validation accuracy of close to 100%. InceptionV3's training and validation loss ranged from 0 to 2. The InceptionRestNetV2 model was 100% accurate during training and validation, and its training and validation losses were close to 0.

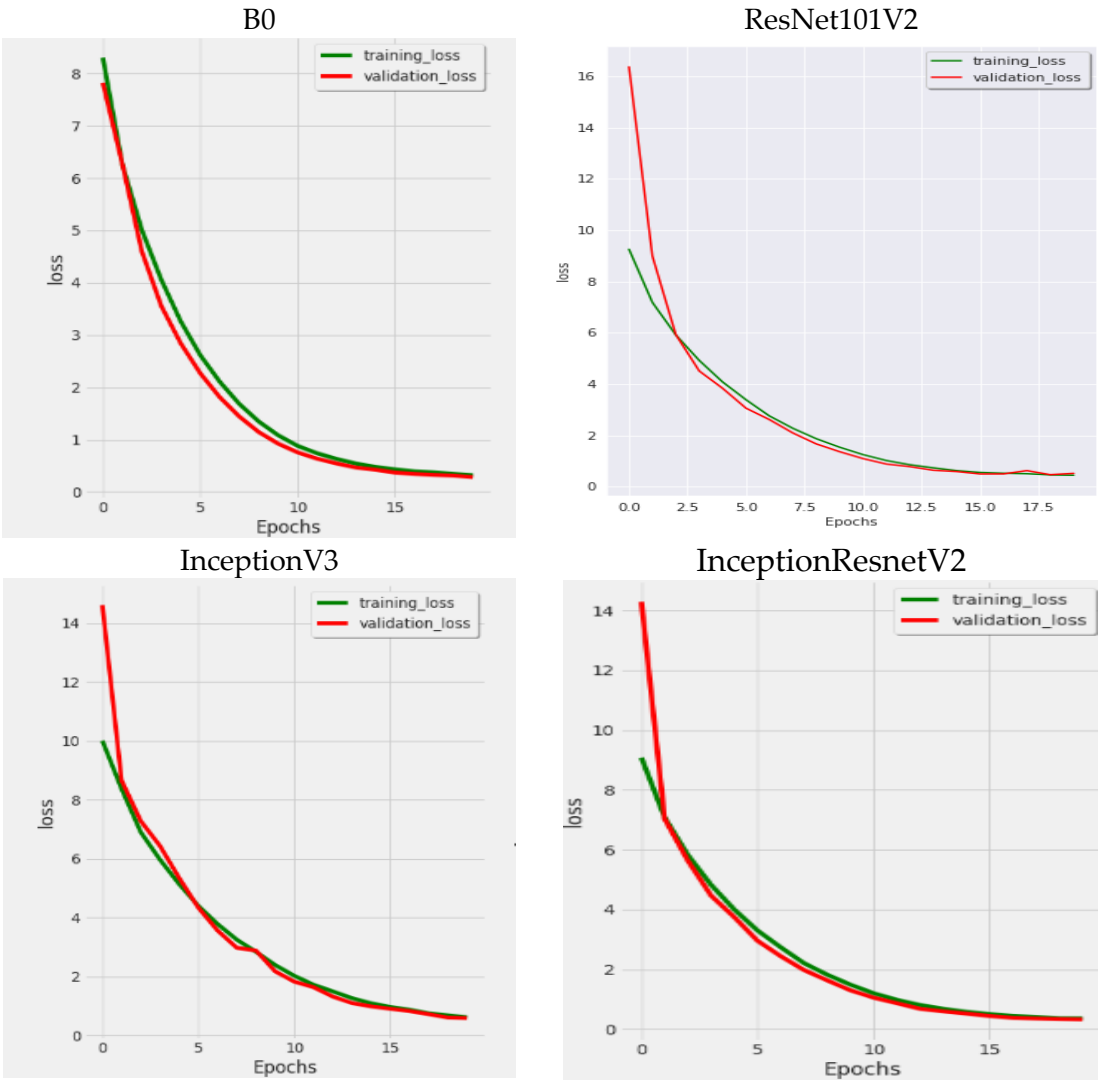

Figure 10. The loss of the four CNN models for each epoch after balancing the Kvasir-Capsule dataset.

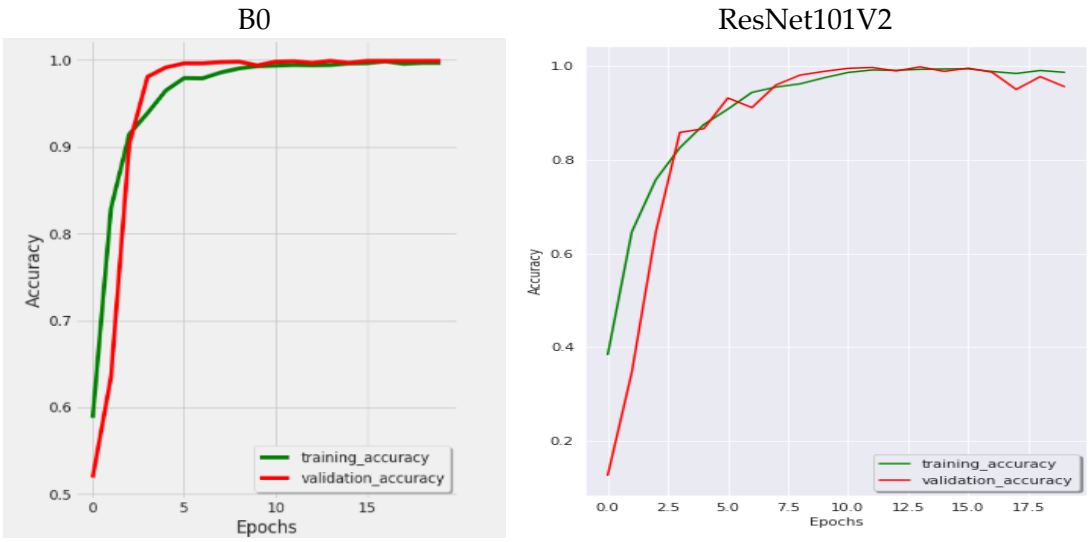

Figure 11. *Cont.*

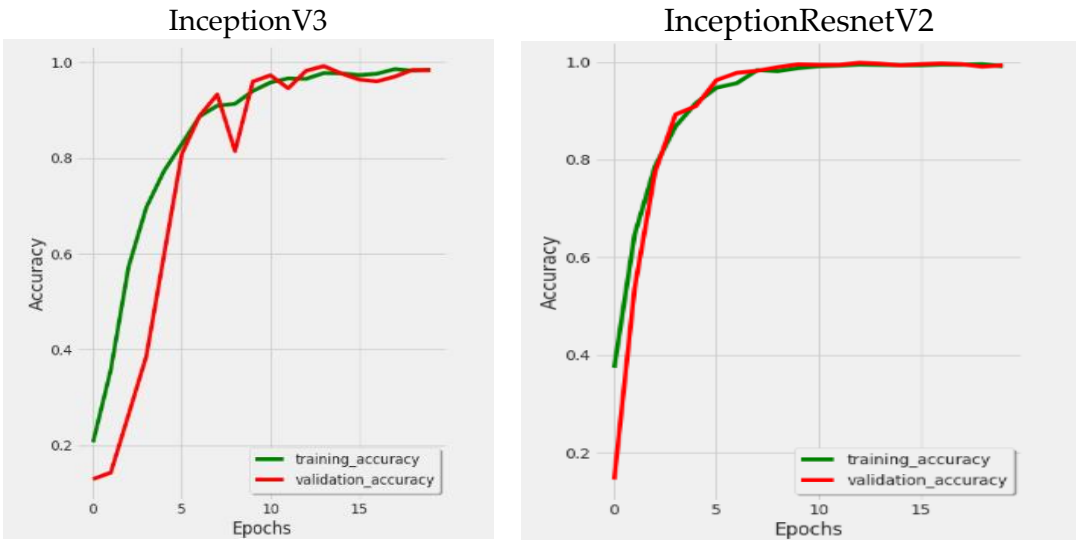

Figure 11. The accuracy of the four CNN models for each epoch after implementing under-sampling and data augmentation on the Kvasir-Capsule dataset.

Figure 12 shows the confusion matrices for the four CNN models in the balanced test set. There were fourteen classes in the test dataset. Ten classes of the test set have 200 images. With an accuracy of 100%, the EfficientNet-B0 model correctly predicted 200 images out of 200 for the class normal mucosa. With an accuracy of 96%, the ResNet101V2 model correctly predicted 192 images. With an accuracy of 99.5%, the InceptionV3 and InceptionResNetV2 models correctly predicted 199 images.

Out of 200 images, the EfficientNet-B0 model correctly predicted 200, achieving an accuracy of 100% for the class ileocecal valve. With an accuracy of 98.5%, the ResNet101V2 model correctly predicted 197 images. With an accuracy of 99.5%, the InceptionV3 and InceptionResNetV2 models correctly predicted 199 images.

The EfficientNet-B0 model correctly predicted 200 images out of 200 images for the class pylorus, with an accuracy of 100%. ResNet101V2 correctly predicted 193 images with an accuracy of 96.5%. The InceptionV3 model correctly predicted 190 images with an accuracy of 95%. The InceptionResNetV2 model correctly predicted 193 images with 96.5 percent accuracy.

4.3. Ablation Study

In the first experiment, we utilized data augmentation techniques on the balanced Kvasir v2 dataset using the static LR and the ILRC processes with four DL models: EfficientNet-B0, ResNet101v2, InceptionV3, and InceptionResNetV2.

Results with Static LR process:

Table 10 presents the results of the first experiment using a static LR with the four DL models. The average accuracies were as follows: EfficientNet-B0 achieved 97.604%, ResNet101v2 reached 96.812%, InceptionV3 obtained 97.562%, and InceptionResNetV2 scored 97.625%.

Table 10. The results of multi-classification for the Kvasir v2 dataset utilizing the static LR.

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
EfficientNet-B0	97.604	90.426	90.488	90.360
ResNet101V2	96.812	87.160	87.961	87.064
InceptionV3	97.562	90.287	90.312	90.163
InceptionResNetV2	97.625	90.559	90.533	90.467

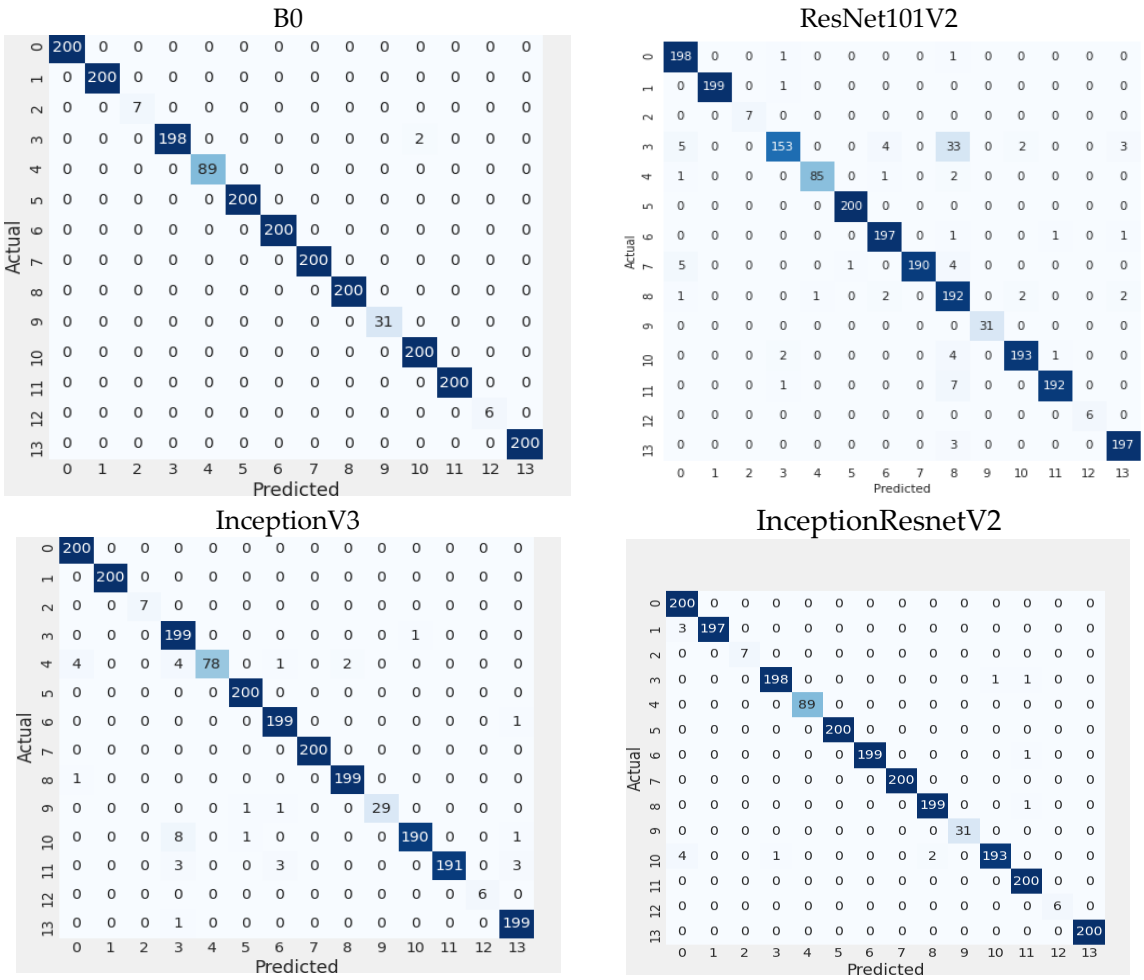

Figure 12. The confusion matrices for the four CNN models after implementing under-sampling and data augmentation on the Kvasir-Capsule dataset.

Results with ILRC Process:

Table 11 displays the results of the first experiment using the ILRC process with the same four DL models. The average accuracies were as follows: EfficientNet-B0 at 97.770%, ResNet101v2 at 97.375%, InceptionV3 at 98%, and InceptionResNetV2 at 98.062%.

Table 11. The results of multi-classification for the Kvasir v2 dataset utilizing the ILRC.

Model	Accuracy (%)	Recall (%)	Precision (%)	F1-Score (%)
EfficientNet-B0	97.770	91.071	91.107	91.061
ResNet101V2	97.375	89.506	89.623	89.471
InceptionV3	98	92.004	92.071	92
InceptionResNetV2	98.062	92.284	92.290	92.242

4.4. Model Complexity

Table 12 shows the initial parameters for the models EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2. It includes their truncated parameters, the number of Floating-Point Operations per Second (FLOPs), training duration, and inference duration. The EfficientNet-B0 model started with approximately 5.3 million parameters before truncation. After truncation, this number decreased to 4.39 million. These parameters were set to a frozen state, meaning they were not re-trained and were used only for feature extraction. Similarly, ResNet101V2 and InceptionV3 also saw significant reductions in their parameter counts. ResNet101V2 decreased from 44.6 million to 43.16 million, while

InceptionV3 dropped from 23.8 million to 22.34 million. InceptionResNetV2 experienced a notable decrease as well, falling from 55.8 million to 54.74 million. Overall, truncation was crucial because it reduced the complexity of the models and streamlined their architecture, which may help lower the risk of overfitting.

Table 12. The four DL models' complexity values (# means number of).

Model/Complexity	# Parameters in Original Model	# Parameters in Proposed Model	Number of FLOPS (G-FLOP)	Training Time (seconds)	Inference Time
EfficientNet-B0	5.3 M	4.39 M	1,856,573,707	1161.97	66 s 94 ms/step
ResNet101V2	44.6 M	43.16 M	32,808,287,188	2085.89	100 s 143 ms/step
InceptionV3	23.8 M	22.34 M	14,373,641,268	1212.79	59 s 84 ms/step
InceptionResNetV2	55.8 M	54.74 M	33,009,236,916	1797.28	122 s 175 ms/step

FLOPs refer to the total number of arithmetic operations involving floating-point numbers (like addition and multiplication) that a system can execute in one second. In greater detail, multiply–add operations are considered two separate FLOPs because, in numerous contemporary models, convolutions do not include a bias term. Therefore, it is logical to classify multiplication and addition as distinct FLOPs.

FLOPs provide an estimate of the computational complexity of the four DL models. Generally, a higher FLOP count suggests a more intricate model, which can result in improved performance but also demands greater computational resources. A model that has fewer FLOPs yet delivers comparable performance to a model with a higher FLOP count is often seen as more efficient. Knowing a model's FLOPs assists in choosing suitable hardware, such as GPUs or TPUs, that can manage the computational workload. Models with high FLOP counts necessitate more powerful hardware for effective operation. In real-time applications, the number of FLOPs directly influences inference speed. Fewer FLOPs can lead to quicker inference times, which is essential for time-sensitive tasks like autonomous driving or real-time video processing. The FLOPs for EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2 were 1,856,573,707; 32,808,287,188; 14,373,641,268; and 33,009,236,916, respectively.

Inference time is an important metric for assessing the complexity of DL models, particularly when implementing them in practical applications. It indicates the duration required for a model to analyze an input and generate an output during the inference phase, which occurs when the model is utilized for making predictions after its training has been completed. The inference times for EfficientNet-B0, ResNet101V2, InceptionV3, and InceptionResNetV2 are 66.094 s per step, 100.143 s per step, 59.084 s per step, 122.175 s per step, and 76.108 s per step, respectively.

4.5. A Comparison of the Model's Results with the Literature Review

Our research introduced the ILRC algorithm, which enhances the training process of DL models, distinguishing it from prior studies. The ILRC algorithm adjusts the LR dynamically based on the training progress, resulting in faster convergence and a reduced risk of overfitting. We evaluated this mechanism across multiple DL architectures, including EfficientNet-B0, ResNet101v2, InceptionV3, InceptionResNetV2, and Xception. Additionally, we improved these models through transfer learning by freezing layers, applying fine-tuning techniques, implementing residual learning, and utilizing regularization strategies. The models were fine-tuned using the Kvasir-Capsule and KVASIR v2 datasets, which feature images obtained from wireless capsule endoscopy (WCE).

The models EfficientNet-B0, ResNet101v2, InceptionV3, InceptionResNetV2, and Xception, when utilizing the ILRC, demonstrated exceptional performance compared to existing state-of-the-art methods, achieving accuracies of 99.906%, 98.765%, 98.312%, 99.344%, and 99.719%, respectively, on the Kvasir-Capsule dataset. On the Kvasir-v2 dataset, they attained accuracies of 97.770%, 97.375%, 98%, 98.0625%, and 98.062%, respectively.

Thus, the ILRC algorithm combined with the EfficientNet-B0, ResNet101v2, InceptionV3, InceptionResNetV2, and Xception models outperformed the most recent methods listed in Table 13 regarding multi-classification accuracy, as shown by the results. Notably, the algorithm achieved an impressive accuracy rate of 99.906%. Moreover, from the patient's perspective, our methodology improves the diagnostic process by reducing costs, expediting diagnosis, and delaying disease progression. By facilitating the multi-classification of GI diseases, the proposed models with the ILRC will automatically identify the disease. As a result, the proposed models utilizing the ILRC will help clinicians detect GI diseases and understand their primary causes. The evaluation of these models showed that, with appropriate tuning, they produce exceptional results compared to the current use of other frameworks.

Table 13. A comparison of the proposed model and state-of-the-art methods.

Reference	Methodology	Accuracy	Dataset
L. Bai et al. [7]	TNU and ViT	79.15% on multi-classification and 98.63% on binary classification	Kvasir-Capsule image and Red Lesion Endoscopy
V. Kumar et al. [15]	Hybrid CNNs	98%	Kvasir-Capsule image
H. Modi et al. [16]	CNN	97.82%	Kvasir-Capsule image
Z. Xiao et al. [17]	WCE-DCGAN	97.25%	Kvasir-Capsule image
S. Mahmood et al. [18]	GIDD-Net and BL-SMOTE	98.9%	Kvasir-Capsule image
A. K. Kundu et al. [13]	KNNs	97.86%	Kvasir-Capsule image
S. Fan et al. [19]	AlexNet	95.34%	Kvasir-Capsule image
S. Charfi et al. [20]	Segmentation, feature descriptor (CLBP + PHOG + Bag of Words (BoW))	94.8%	WEO clinical endoscopy atlas and Kvasir Capsule endoscopy
A. Caroppo et al. [21]	VGG19, InceptionV3, ResNet50, and SVM	95.70%	Kvasir-Capsule image and Medical Image Computing and Computer-Assisted Intervention (MICCAI) 2017
H. Gunasekaran et al. [22]	DenseNet201, InceptionV3, and ResNet50	95%	Kvasir v2
The Proposed Models	EfficientNet-B0, ResNet101v2, InceptionV3, InceptionResNetV2, and Xception utilizing the ILRC	**99.906% and 98.0625%**	**Kvasir-Capsule image and Kvasir-v2 datasets**

5. Conclusions

In this study, we presented the ILRC mechanism aimed at enhancing the training efficiency of DL models. The ILRC dynamically modifies the LR in response to the training process's advancements, which accelerates convergence and minimizes the risk of overfitting. We evaluated this mechanism across various DL architectures, including EfficientNet-B0, ResNet101v2, InceptionV3, InceptionResNetV2, and Xception. Additionally, we improved the model layers by combining traditional transfer learning with freezing layers, fine-tuning strategies, residual learning, and contemporary regularization techniques. These models were fine-tuned using the Kvasir-Capsule and KVASIR v2 datasets, which comprise images sourced from WCE.

To address the class imbalance within the Kvasir-Capsule dataset, we employed an under-sampling method to achieve a balanced distribution of images among the classes. Furthermore, we utilized a data augmentation strategy (over-sampling) to create additional WCE images from existing samples, effectively enlarging both datasets. This process involved altering four attributes of the existing WCE images. We also resized the images in the two datasets to 336 × 336 pixels and normalized their contrast.

The models—EfficientNet-B0, ResNet101v2, InceptionV3, InceptionResNetV2, and Xception—demonstrated remarkable performance compared to existing benchmarks, at-

taining accuracies of 99.906%, 98.765%, 98.312%, 99.344%, and 99.719%, respectively, on the Kvasir-Capsule dataset. On the Kvasir-v2 dataset, they achieved accuracies of 97.770%, 97.375%, 98%, 98.0625%, and 98.062%, respectively. The combination of the ILRC with traditional transfer learning, freezing layers, fine-tuning methods, residual learning, and modern regularization techniques in our proposed models provides a strong solution for automating the detection of gastrointestinal abnormalities in WCE images, thereby enhancing diagnostic efficiency and accuracy in clinical practice.

The limitations of our proposed model are related to the training and testing phases. The structure of the model contributed to this issue. Therefore, we plan to simplify our model in future work to enhance the speed of the training and testing phases. Additionally, we will test the proposed model on different types of human diseases and use hyper-optimization algorithms to automatically improve hyper-parameterization.

Author Contributions: Conceptualization, S.A.E.-G. and A.A.A.E.-A.; methodology, S.A.E.-G.; software, A.A.A.E.-A. and M.A.M.; validation, S.A.E.-G., A.A.A.E.-A. and M.A.M.; formal analysis, S.A.E.-G.; investigation, A.A.A.E.-A.; resources, M.A.M.; data curation M.A.M.; writing—original draft preparation, A.A.A.E.-A.; writing—review and editing, S.A.E.-G.; visualization M.A.M.; supervision, S.A.E.-G.; project administration, S.A.E.-G.; funding acquisition, S.A.E.-G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Deanship of Graduate Studies and Scientific Research at Jouf University under grant No. (DGSSR-2023-02-02416).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The datasets used in this study are publicly available and can be accessed from their respective sources: The first dataset used in this article was obtained from the Open Science Framework website (OSF), which is freely available for all researchers and scientists to conduct experiments. You can access it at the following website: https://osf.io/dv2ag/ (accessed on 3 November 2024). The second dataset, the Kvasir-v2 dataset, is available at simula and can be accessed from https://datasets.simula.no/kvasir/ (accessed on 3 November 2024).

Acknowledgments: The authors extend their appreciation to the Deanship of Graduate Studies and Scientific Research at Jouf University for funding this work through research grant No. (DGSSR-2023-02-02416).

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

References

1. Smedsrud, P.H.; Thambawita, V.; Hicks, S.A.; Gjestang, H.; Nedrejord, O.O.; Næss, E.; Borgli, H.; Jha, D.; Jan Derek Berstad, T.; Eskeland, S.L.; et al. Kvasir-Capsule, a video capsule endoscopy dataset. *Sci Data* **2021**, *8*, 142. [CrossRef] [PubMed]
2. WHO. Gastrointestinal Cancer. Available online: https://www.who.int/news-room/fact-sheets/detail/cancer (accessed on 25 March 2022).
3. Shahril, R.; Baharun, S.; Islam, A. Pre-processing technique for wireless capsule endoscopy image enhancement. *Int. J. Electr. Comput. Eng.* **2016**, *6*, 1617–1626.
4. Amiri, Z.; Hassanpour, H.; Beghdadi, A. A computer-aided method to detect bleeding frames in capsule Endoscopy images. In Proceedings of the 8th European Workshop on Visual Information Processing, Roma, Italy, 28–31 October 2019.
5. Yuan, Y.; Li, B.; Meng, M.Q.H. Bleeding frame and region detection in the Wireless Capsule Endoscopy video. *IEEE J. Biomed. Health Inf.* **2015**, *20*, 624–630. [CrossRef] [PubMed]
6. Shamsudhin, N.; Zverev, V.I.; Keller, H.; Pane, S.; Egolf, P.W.; Nelson, B.J.; Tishin, A.M. Magnetically guided capsule endoscopy. *Med. Phys.* **2017**, *44*, e91–e111. [CrossRef]
7. Bai, L.; Wang, L.; Chen, T.; Zhao, Y.; Ren, H. Transformer-based disease identification for small-scale imbalanced capsule endoscopy dataset. *Electronics* **2022**, *11*, 2747. [CrossRef]
8. Shen, L.; Shan, Y.S.; Hu, H.M.; Price, T.J.; Sirohi, B. Management of gastric cancer in Asia: Resource-stratified guidelines. *Lancet Oncol.* **2013**, *14*, e535–e547. [CrossRef]
9. Liao, Z.; Hou, X.; Lin-Hu, E.Q.; Sheng, J.Q.; Ge, Z.Z.; Jiang, B.; Hou, X.-H.; Liu, J.-Y.; Li, Z.; Huang, Q.-Y.; et al. Accuracy of magnetically controlled capsule endoscopy, compared with conventional gastroscopy, in detection of gastric diseases. *Clin. Gastroenterol. Hepatol.* **2016**, *14*, 1266–1273. [CrossRef]

10. Ozyoruk, K.B.; Gokceler, G.I.; Bobrow, T.L.; Coskun, G.; Incetan, K.; Almalioglu, Y.; Mahmood, F.; Curto, E.; Perdigoto, L.; Oliveira, M.; et al. EndoSLAM dataset and an unsupervised monocular visual odometry and depth estimation approach for endoscopic videos. *Med. Image Anal.* **2021**, *71*, 102058. [CrossRef] [PubMed]
11. Stewart, B.J.; Ferdinand, J.R.; Clatworthy, M.R. Using single-cell technologies to map the human immune system—Implications for nephrology. *Nat. Rev. Nephrol.* **2020**, *16*, 112–128. [CrossRef]
12. Deeba, F.; Mohammed, S.K.; Bui, F.M.; Wahid, K.A. A saliency-based unsupervised method for angiectasia detection in endoscopic video frames. *J. Med. Biol. Eng.* **2018**, *38*, 325–335. [CrossRef]
13. Kundu, A.K.; Fattah, S.A.; Rizve, M.N. An automatic bleeding frame and region detection scheme for wireless capsule endoscopy videos based on interplane intensity variation profile in normalized RGB color space. *J. Healthc. Eng.* **2018**, *2018*, 9423062. [CrossRef] [PubMed]
14. Hong, T.C.; Liou, J.M.; Yeh, C.C.; Yen, H.H.; Wu, M.S.; Lai, I.-R.; Chen, C.-C. Endoscopic submucosal dissection comparing with surgical resection in patients with early gastric cancer—A single center experience in Taiwan. *J. Formos. Med. Assoc.* **2020**, *119*, 1750–1757. [CrossRef] [PubMed]
15. Kumar, V.; Jain, S.; Singh, N. A Hybrid Convolutional Neural Network with Meta Feature Learning for Abnormality Detection in Wireless Capsule Endoscopy Images. *arXiv* **2022**, arXiv:2207.09769.
16. Modi, H.; Misra, S.; Gaur, S. Digestive tract abnormalities classification using wireless capsule endoscopy data. *Int. J. Innov. Sci. Res. Technol.* **2021**, *6*, 505–509.
17. Xiao, Z.; Lu, J.; Wang, X.; Li, N.; Wang, Y.; Zhao, N. WCE-DCGAN: A data augmentation method based on wireless capsule endoscopy images for gastrointestinal disease detection. *IET Image Process.* **2022**, *17*, 1170–1180. [CrossRef]
18. Mahmood, V.; Fareed, M.M.S.; Ahmed, G.; Dawood, F.; Zikria, S. A robust deep model for classification of peptic ulcer and other digestive tract disorders using endoscopic images. *Biomedicines* **2022**, *10*, 2195. [CrossRef]
19. Fan, S.; Xu, L.; Fan, Y.; Wei, K.; Li, L. Computer-aided detection of small intestinal ulcer and erosion in wireless capsule endoscopy images. *Phys. Med. Biol.* **2018**, *63*, 165001. [CrossRef]
20. Charfi, S.; El Ansari, M.; Balasingham, I. Computer-aided diagnosis system for ulcer detection in wireless capsule endoscopy images. *IET Image Process.* **2019**, *13*, 1023–1030. [CrossRef]
21. Caroppo, A.; Leone, A.; Siciliano, P. Deep transfer learning approaches for bleeding detection in endoscopy images. *Comput. Med. Imaging Graph.* **2020**, *88*, 101852. [CrossRef]
22. Gunasekaran, H.; Ramalakshmi, K.; Swaminathan, D.K.; Mazzara, M. GIT-Net: An Ensemble Deep Learning-Based GI Tract Classification of Endoscopic Images. *Bioengineering* **2023**, *10*, 809. [CrossRef]
23. Yogapriya, J.; Chandran, V.; Sumithra, M.; Anitha, P.; Jenopaul, P.; Dhas, C.S.G. Gastrointestinal tract disease classification from wireless endoscopy images using pretrained deep-learning model. *Comput. Math. Methods Med.* **2021**, *2021*. [CrossRef] [PubMed]
24. Pogorelov, K.; Randel, K.R.; Griwodz, C.; Eskeland, S.L.; de Lange, T.; Johansen, D.; Spampinato, C.; Dang-Nguyen, D.-T.; Lux, M.; Schmidt, P.T.; et al. KVASIR: A multi-class image dataset for computer-aided gastrointestinal disease detection. In Proceedings of the 8th ACM on Multimedia Systems Conference, New York, NY, USA, 20–23 June 2017; pp. 164–169.
25. Saponara, S.; Elhanashi, A. Impact of image resizing on deep learning detectors for training time and model performance. In *ApplePies, Lecture Notes in Electrical Engineering*; Springer: Berlin/Heidelberg, Germany, 2022; Volume 866.
26. Yosinski, J.; Clune, J.; Bengio, Y.; Lipson, H. How transferable are features in deep neural networks? *Adv. Neural Inf. Process. Syst. (NeurIPS)* **2014**, *27*, 3320–3328.
27. Goodfellow, I.; Shlens, J.; Szegedy, C. Explaining and harnessing adversarial examples. In Proceedings of the International Conference on Learning Representations (ICLR), San Diego, CA, USA, 7–9 May 2015.
28. Hasan, T.; Alexandro, G.; Julian, H.; Thomas, T.; Christian, H.; Tobias, M. Transfer-Learning: Bridging the Gap between Real and Simulation Data for Machine Learning in Injection Molding. *Procedia CIRP* **2018**, *72*, 85–190.
29. Bilal, M.; Maqsood, M.; Yasmin, S.; Hasan, N.U.; Rho, S.A. Transfer learning-based efficient spatiotemporal human action recognition framework for long and overlapping action classes. *J. Supercomput.* **2022**, *78*, 2873–2908. [CrossRef]
30. Tan, M.; Le, Q. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning, Long Beach, CA, USA, 9–15 June 2019; pp. 6105–6114.
31. Alhichri, H.; Alswayed, A.S.; Bazi, Y.; Ammour, N.; Alajlan, N.A. Classification of remote sensing images using efficientnet-b3 cnn model with attention. *IEEE Access* **2021**, *9*, 14078–14094. [CrossRef]
32. Sandler, M.; Howard, A.; Zhu, M.; Zhmoginov, A.; Chen, L.-C. MobileNetV2: Inverted residuals and linear bottlenecks. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 4510–4520.
33. Putra, A.; Rufaida, S.I.; Leu, J. Enhanced skin condition prediction through machine learning using dynamic training and testing augmentation. *IEEE Access* **2020**, *4*, 40536–40546. [CrossRef]