

Article

Deep Learning and Vision-Based Early Drowning Detection

Maad Shatnawi , Frdoos Albreiki, Ashwaq Alkhoori and Mariam Alhebshi

Department of Electrical Engineering Technology, Higher Colleges of Technology, Abu Dhabi P.O. Box 25035, United Arab Emirates

* Correspondence: mshatnawi@hct.ac.ae

Abstract: Drowning is one of the top five causes of death for children aged 1–14 worldwide. According to data from the World Health Organization (WHO), drowning is the third most common reason for unintentional fatalities. Designing a drowning detection system is becoming increasingly necessary in order to ensure the safety of swimmers, particularly children. This paper presents a computer vision and deep learning-based early drowning detection approach. We utilized five convolutional neural network models and trained them on our data. These models are SqueezeNet, GoogleNet, AlexNet, ShuffleNet, and ResNet50. ResNet50 showed the best performance, as it achieved 100% prediction accuracy with a reasonable training time. When compared to other approaches, the proposed approach performed exceptionally well in terms of prediction accuracy and computational cost.

Keywords: drowning detection; drowning rescue; swimming pool surveillance; AI; deep learning; machine learning; computer vision; CNN; convolution neural networks



Citation: Shatnawi, M.; Albreiki, F.; Alkhoori, A.; Alhebshi, M. Deep Learning and Vision-Based Early Drowning Detection. *Information* **2023**, *14*, 52. <https://doi.org/10.3390/info14010052>

Academic Editors: Danilo Avola, Daniele Pannone and Alessio Fagioli

Received: 25 October 2022

Revised: 22 December 2022

Accepted: 29 December 2022

Published: 16 January 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

According to the World Health Organization (WHO) report [1], drowning is the third most common reason for unintentional fatalities worldwide for children and young people aged 1–14 years, with children under the age of 5 at highest risk. There are an estimated 236,000 annual drowning deaths around the world [2]. In 48 of the 85 countries, drowning is one of the top five fatalities of children between the ages of 1 and 14 [3]. According to [4], as the population increases and the development of hotels and villas with swimming pools becomes more popular, the death rate due to drowning will increase. Several investigations have been performed by governments and organizations to find appropriate ways to save people. Some of these ways include providing information on dangers of drowning given to parents through the child surveillance programs, encouraging fencing or draining of garden ponds and domestic swimming pools, and increasing supervision of swimming in lakes, rivers, and beaches in order to reduce the number of accidents. Unfortunately, these solutions are not enough and can be considered rudimentary. The effective reduction in drowning and the assurance of pool safety can be achieved through the implementation of a smart automated monitoring system.

There are several approaches for automatic drowning detection which can be categorized into two classes. The approaches of the first category are based on wearing sensing devices that are attached to the swimmer through a wristband or goggles. These sensors can monitor the swimmer behavior through providing measurements such as heart rate, blood oxygen level, motion, hydraulic pressure, and depth. The second category involves vision-based approaches where overhead or underwater cameras are used to monitor the swimmers, and machine learning (ML) algorithms are employed to detect drowning instances from the output of these cameras.

The main contribution of this study is the proposal of a ground-breaking technique that quickly and automatically detects drowning victims based on deep learning convolutional neural networks (CNNs). We investigated five pretrained CNN models for identifying drowning cases within a swimming pool. These models were AlexNet [5], GoogleNet [6],

SqueezeNet [7], ShuffleNet [8], and ResNet [9]. After fine-tuning these models and training them on our dataset, all models successfully identified the drowning instances from normal swimming events with a very high prediction accuracy and confidence level.

2. Related Work

Automatic drowning detection approaches can be classified as sensor-based and ML-based approaches. There are very few available ML-based drowning detection approaches in the literature. Alotaibi [10] proposed a swimming pool monitoring system based on IoT and transfer learning. The motion sensor detects any objects around the swimming pool and sends a signal to an overhead camera which captures a single image. The image is sent via wireless communication to the server station, for processing and classification. The ResNet50 model classifies the detected object as human, animal, or object. Li et al. [11] proposed a technique for identifying drowning victims at sea. They created a dataset of 6079 images using a team of actors. They employed the Yolov3 algorithm with some modifications. Briefly, the residual module with channel attention mechanism was used in the feature extraction network, a bottom-up structure was added to the feature fusion network (FPN) structure, CIoU was used as a loss function, and a linear transformation method was used to deal with the anchor boxes generated by a clustering algorithm. The model classifies human targets into four categories: sea person, uncertain sea person, land person, and uncertain land person. The model achieved an accuracy of 72.17%.

Chan et al. [12] presented an AlexNet CNN model with the use of NVIDIA Jetson Nano. The model was trained using 1168 drowning images and 2333 non-drowning images. The testing dataset contained 389 drowning images and 777 non-drowning images. The dataset was created by 30 volunteers who made different poses in the pool. The model achieved 85% classification accuracy. Handalage et al. [13] proposed a drowning rescue system with three main functions: detecting drowning victims, sending drones to victims, and detecting dangerous activities. The drowning detection component detects drowning victims through a CNN model. The second component is the rescue drone which is sent to the victim's location coordinates. The third component detects dangerous activities such as running around the swimming pool and drinking. The drowning detection system was trained through 5000 images representing four categories: drowning stage 1, drowning stage 2, drowning stage 3, and not drowning. The major source of the data was the introduction of actors and the collection of videos in real time. The secondary source of data was the Internet. Swimmers in the pool were detected using an overhead camera. YOLO [14] was used to detect objects by locating one or more objects in the image and sorting each object. The CNN model was implemented on the NVIDIA Jetson Nano board in order to run multiple neural models in parallel.

Hasan et al. (2021) [15] presented a video dataset collected by overhead and underwater cameras, including three water activity behaviors which are swim, drown, and idle. The dataset contained 47 overhead videos and 44 underwater videos, resulting in 24,729 and 22,010 video frames, respectively. Three pretrained CNN models, ResNet50 [9], VGG16 [16], and MobileNet [17], were evaluated on this dataset. These models achieved a detection accuracy of 96.85%, 83.25%, and 96.7% respectively.

3. Materials and Methods

The main contribution of this study is the development of an automated and intelligent system for monitoring swimming pools for early drowning detection. We utilize deep machine learning to efficiently process the swimmers' images and enable early detection of any drowning case.

3.1. Dataset

Our dataset contains 200 images that were collected through the Google search engine. The data consist of two classes (drowning and swimming), where each class includes 100 images representing swimmers from both genders but different ages. We used 100 of

these images (50 drowning and 50 swimming) for model training and validation, while the other 100 images (50 drowning and 50 swimming) were used for model testing. The training and validation dataset was further split into 70% for training and 30% for validation. Samples of the drowning and swimming images are presented in Figure 1.

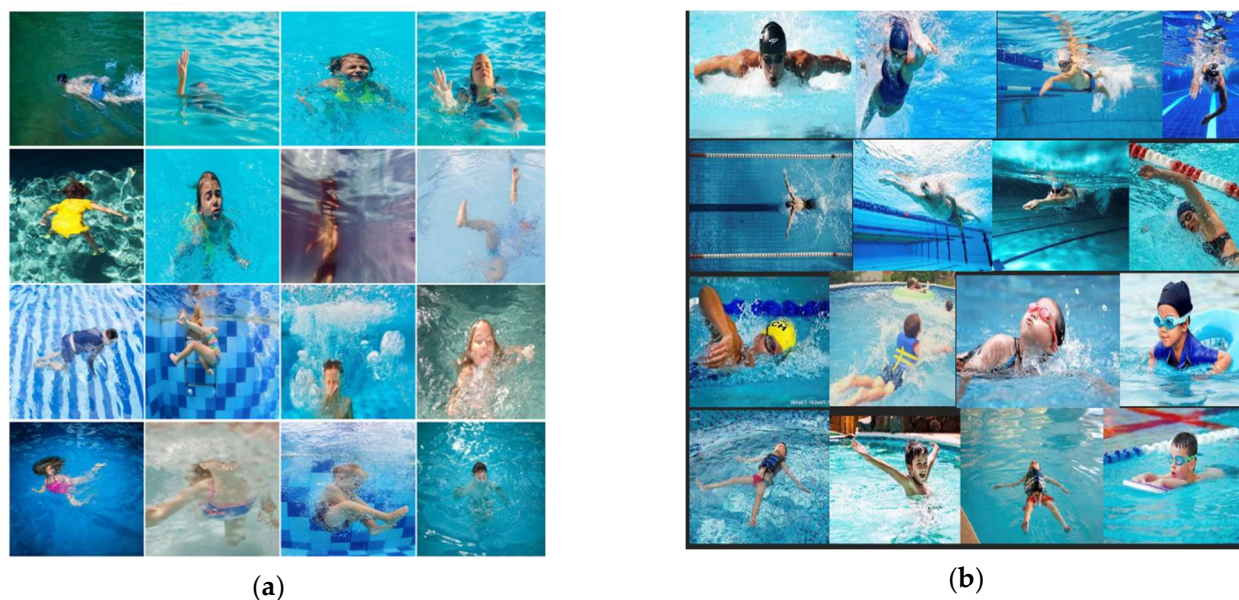


Figure 1. (a) Samples of drowning data; (b) samples of swimming data.

Machine learning algorithms usually require large training data to perform well. In our case, the available data were limited, which may cause overfitting problems. To address this issue, we utilized data augmentation and transfer learning. Data augmentation is a mechanism for increasing the quantity of data by introducing slightly modified copies of current data or newly created synthetic data from existing data [18]. It regularizes and aids in the training of a machine learning model to reduce overfitting. Data augmentation in deep learning takes the form of geometric modifications, flipping, color alteration, cropping, rotation, noise injection, and random erasure to improve the image [19].

After loading our data into the network, we used two forms of data augmentation: rotation and scaling. For each CNN, we used different rotational angles. For GoogleNet, ShuffleNet, AlexNet, and ResNet50 we used a rotational angle of -45° to 45° . After many trials and errors, this rotational angle achieved the highest accuracy for these four CNNs. However, the rotational angle for SqueezeNet that achieved the highest validation accuracy was -60° to 60° . On each image, random scaling factors in the range of 1 to 2 were applied for all five CNNs.

The experiments in this work were implemented using the deep learning toolbox in MATLAB where rotation angles and scale factors are picked randomly from continuous uniform distributions within the specified intervals. Each epoch produces slightly different transformed versions of each image in the training dataset while maintaining an equal number of training images across epochs. These transformed images are not stored in memory [20].

3.2. CNN Models

Deep learning algorithms including CNN have led to significant advances in the field of computer vision in recent years. The major advantage of a CNN is that it can learn directly from input images, eliminating the need for preprocessing and feature extraction techniques [21,22].

Three of the most critical characteristics that are typically considered when selecting a convolutional neural network model are classification accuracy, computational time, and

memory requirement [23]. Due to time constraints, computational limitations, and the unavailability of an adequate amount of training data, which often make it a significant challenge to build a CNN model from scratch, using pretrained CNN models is considered a good option. There are several publicly available pretrained CNN models [19]. In this work, we examined five pretrained networks; AlexNet, GoogleNet, SqueezeNet, ShuffleNet, and ResNet50. These models were selected in this study as they have been successfully implemented in many state-of-the-art research publications and have shown great performance in several applications.

AlexNet was one of the first deep convolutional networks that reached significant accuracy [5]. The overfitting problem is solved in AlexNet by using dropout layers, where a connection is dropped with a probability of 0.5 during testing. A probability of 0.5 was chosen since it was the best fit for the network parameters and training options. Although this prevents the network from overfitting by allowing it to escape from undesirable local minima, it also doubles the number of iterations required for convergence. Millions of images have been classified using this algorithm into object categories, such as faces, fruit, cups, pencils, and animals. Networks take images as input and assign labels for those objects. In addition, they take probabilities for the categories in which those objects fall. There are two sets of images involved with the input to the network: $227 \times 227 \times 3$ RGB images [24–27]. The AlexNet architecture is shown in Figure 2.

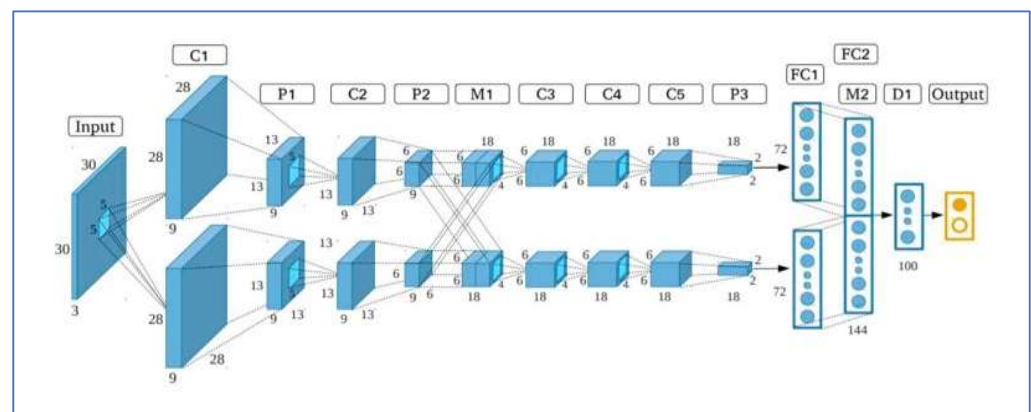


Figure 2. AlexNet architecture [27].

The inception module in the GoogleNet design solved most of the problems that huge networks faced [6]. GoogleNet has an error rate of 6.67%, which is very close to human performance. The design consists of 22 deep CNN layers, lowering the number of parameters to four million (60 million compared to AlexNet). In addition to the 22 layers of GoogleNet, there are five pooling layers [28]. The initiation modules comprise nine linear layers altogether. There are also 1×1 convolution filters. In part due to the parallel network implementation and layer reduction, the network has very good computational and memory efficiency. The model size is also smaller than other networks [24,27]. The GoogleNet architecture is presented in Figure 3.

SqueezeNet is a small CNN requiring less communication between servers during distribution training [7]. Smaller CNNs are also easier to implement on hardware with limited memory, such as a field-programmable gate array (FPGA). SqueezeNet is a convolutional neural network with 18 layers. Image categories are categorized into 1000 categories by the pretrained network. The network learns complex function representations for a wide variety of images. The goal of utilizing SqueezeNet is to create a smaller neural network using fewer datasets that can be readily integrated into computer memory and transmitted via a computer network [24,27,29]. The SqueezeNet architecture is shown in Figure 4.

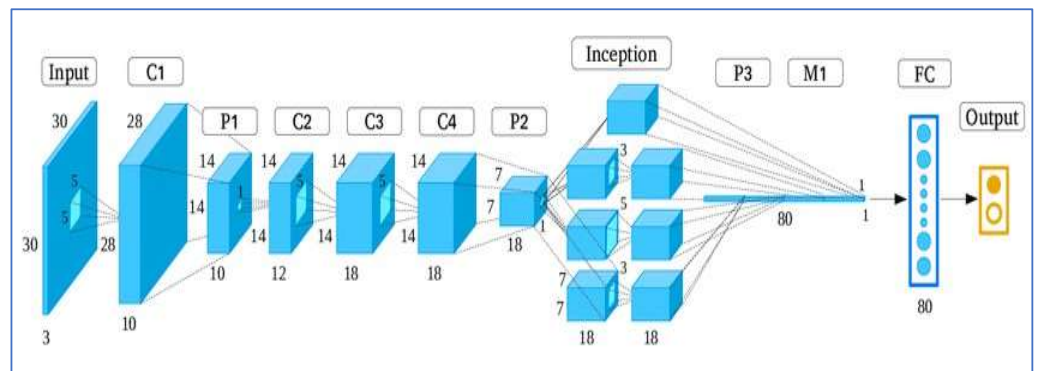


Figure 3. GoogleNet architecture [27].

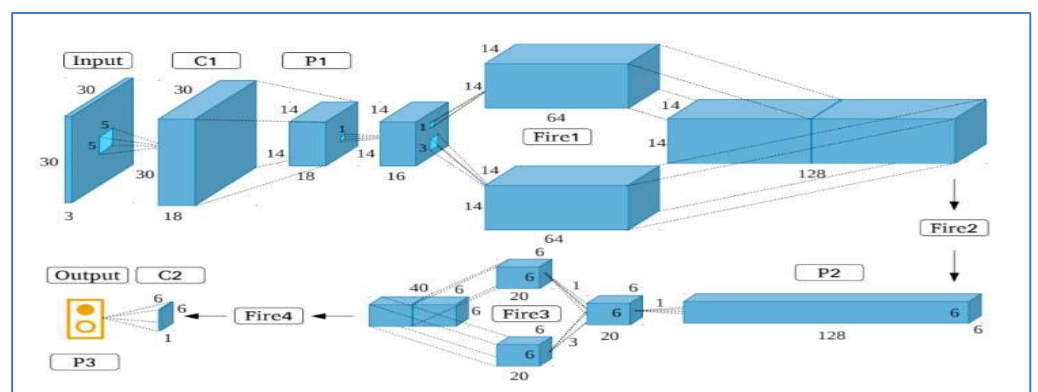


Figure 4. SqueezeNet architecture [27].

ShuffleNet is a very resource-efficient CNN architecture that was created specifically for mobile devices with very little processing power [8]. The network architecture significantly lowers computation costs while retaining accuracy by using two new operations, pointwise group convolution and channel shuffle. The ShuffleNet architecture is illustrated in Figure 5.

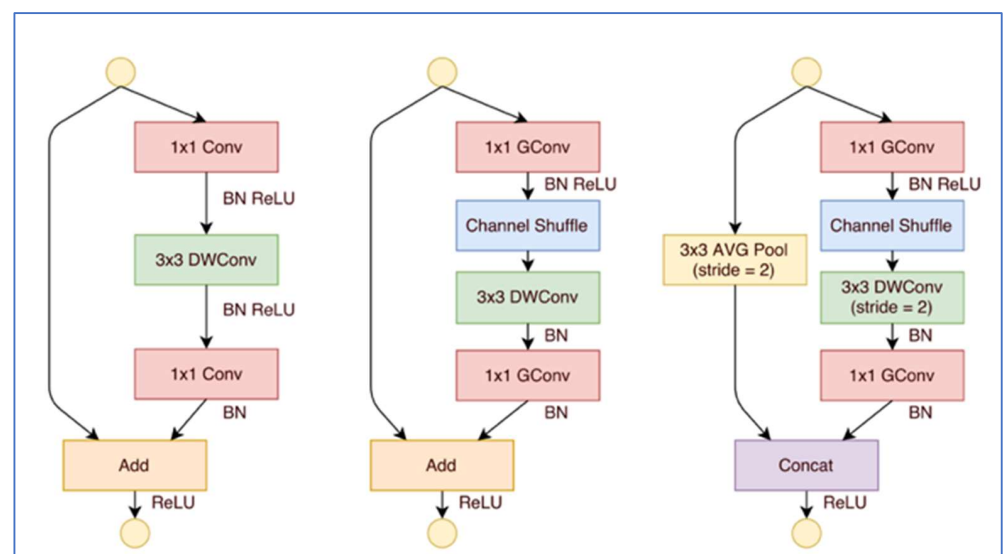


Figure 5. ShuffleNet architecture [30].

ResNet was developed in 2016 by introducing some features that significantly increase network accuracy and speed [9]. Not every neuron in the ResNet design needs to fire at

once. After learning a feature once, it does not try to learn it again; instead, it focuses on learning additional features. This strategy enhances the effectiveness of model training. The ResNet50 network consists of 50 layers, and its architecture is illustrated in Figure 6.

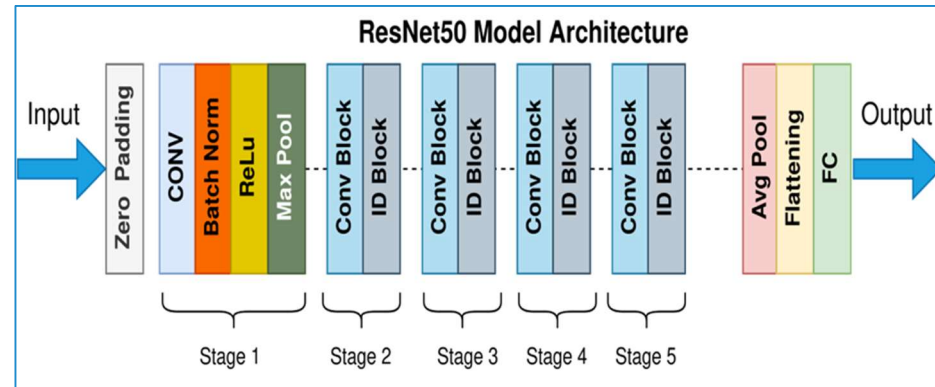


Figure 6. ResNet50 architecture [31].

When training any pretrained network, we start by modifying the parameters of the basic design. In each of the CNN models, we tuned the pretrained network parameters, i.e., the convolution 2D layer and the classification output layer. We modified the filter size to 1×1 , and the number of filters to 2 as we had two classes of data. We modified the classification output layer to suit our output classification and labels. Furthermore, we set the starting learning rate to 0.0001, the validation frequency to 5, and the maximum epochs to 60, since we wanted to avoid the failure of training pauses based on error rates. An epoch is a single learning cycle in which the learner is exposed to the whole training dataset. Furthermore, the minimum batch size is equal to 11, which corresponds to the memory needs (8.00 GB) of the CPU hardware, which operates at 1.8 GHz. Increasing the number of epochs leads to better training and validation accuracy, but it may lead to overfitting. The training dataset was randomly separated into two parts: 70% of the data for training and 30% of the data for validation to avoid overfitting.

3.3. Evaluation Measures

A machine learning model can be assessed and compared to other methods using a variety of performance metrics. The most commonly used evaluation metrics are accuracy, sensitivity, specificity, precision, F1, and MCC. The accuracy (Ac) metric measures the percentage of correctly predicted drowning and swimming instances in the testing dataset. The percentage of drowning instances that were successfully predicted relative to all of the drowning cases included in the dataset is known as sensitivity or recall (R). Precision (Pr) measures the percentage of accurately predicted drowning cases to all predicted drowning cases. The percentage of accurately predicted non-drowning cases to all non-drowning cases listed in the dataset is known as specificity (Sp). These metrics can be mathematically represented as follows [32,33]:

$$Ac = \frac{(TP + TN)}{(TP + TN + FN + FP)} \quad (1)$$

$$R = \frac{TP}{(TP + FN)} \quad (2)$$

$$Pr = \frac{TP}{(TP + FP)} \quad (3)$$

$$Sp = \frac{TN}{(TN + FP)} \quad (4)$$

where TP, TN, FP, and FN represent true positive, true negative, false positive, and false negative instances, respectively.

The F-measure (F1) is an evaluation metric that integrates precision and recall into a single value. A statistic that strikes a compromise between prediction sensitivity and specificity is the Mathew correlation coefficient (MCC). MCC is a numeric scale that goes from -1 , which denotes an inverse prediction, through 0 , which stands for a random classifier, to $+1$, which denotes a flawless prediction [34–36].

$$F1 = \frac{2PR}{P + R} \quad (5)$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FN)(TP + FP)(TN + FP)(TN + FN)}} \quad (6)$$

4. Results and Discussion

4.1. Model Training and Validation

We examined five different pretrained CNN models. We started with GoogleNet training, as shown in Figure 7, which required a total of 60 epochs with two iterations per each epoch for a total of 120 iterations for the network to properly train and validate the data. It achieved a validation accuracy of 91.67% after 120 iterations. The network training took 3 min and 36 s to complete. Furthermore, the validation was carried out in a five-iteration process to verify that the system was well trained, while avoiding overfitting of the data.

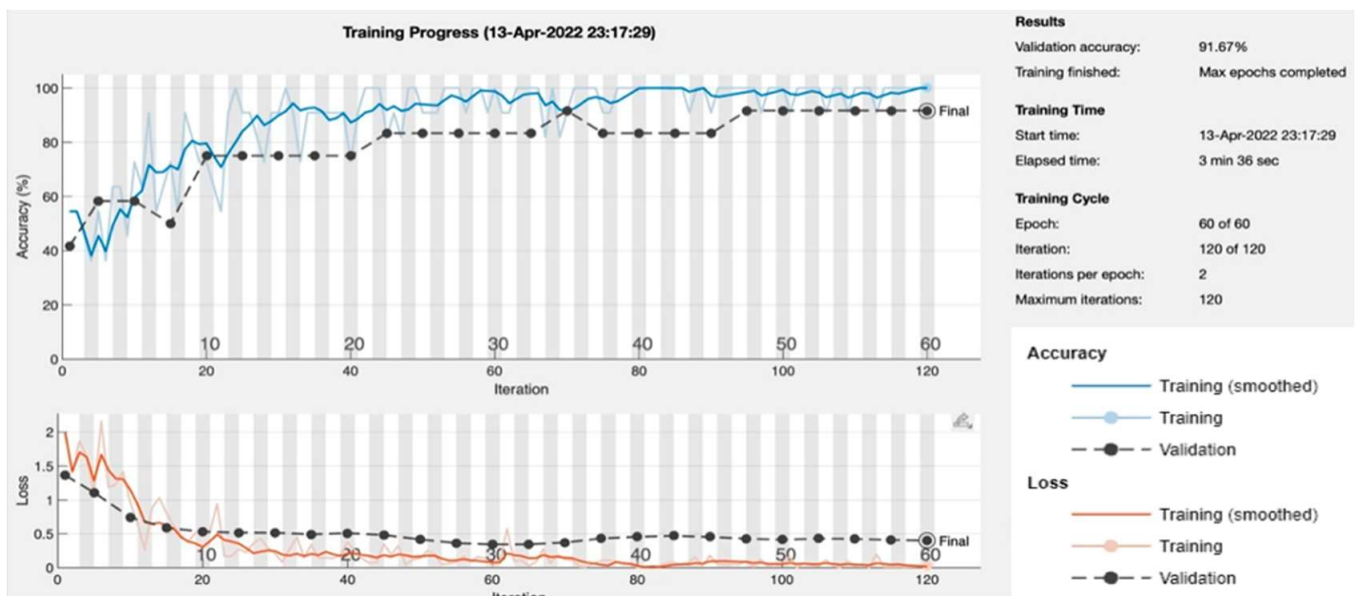


Figure 7. GoogleNet performance results.

To train SqueezeNet, we used a total of 20 epochs with two iterations per each epoch, as shown in Figure 8. The model achieved a validation accuracy of 100% after 40 iterations, which is ideal accuracy and indicates a well-trained network. The training procedure took only 26 s, which was significantly less time than the other two networks. To avoid data overfitting, the validation frequency was also performed in a five-iteration process.

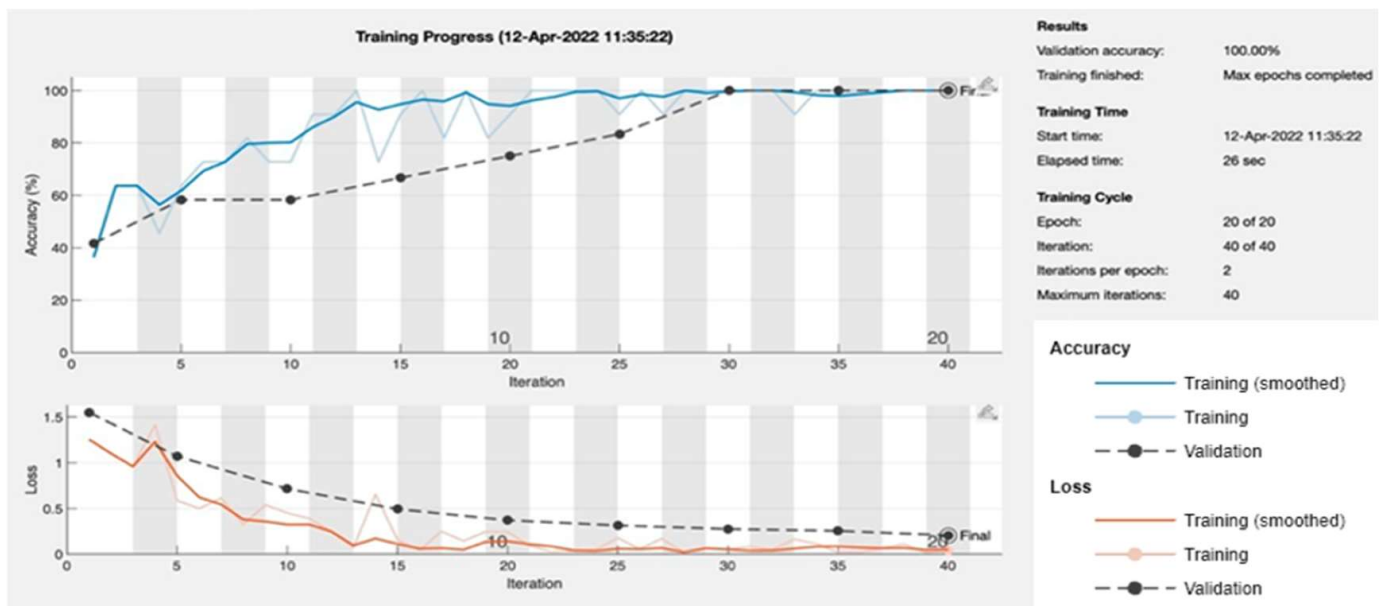


Figure 8. The training performance of SqueezeNet.

However, there were 15 epochs overall for training AlexNet (Figure 9), with two iterations per epoch, allowing the network to train and validate the data extremely successfully. It attained a validation accuracy of 91.67% after 30 iterations. The network training took 1 min and 5 s to complete, and the validation was carried out through a five-iteration process.

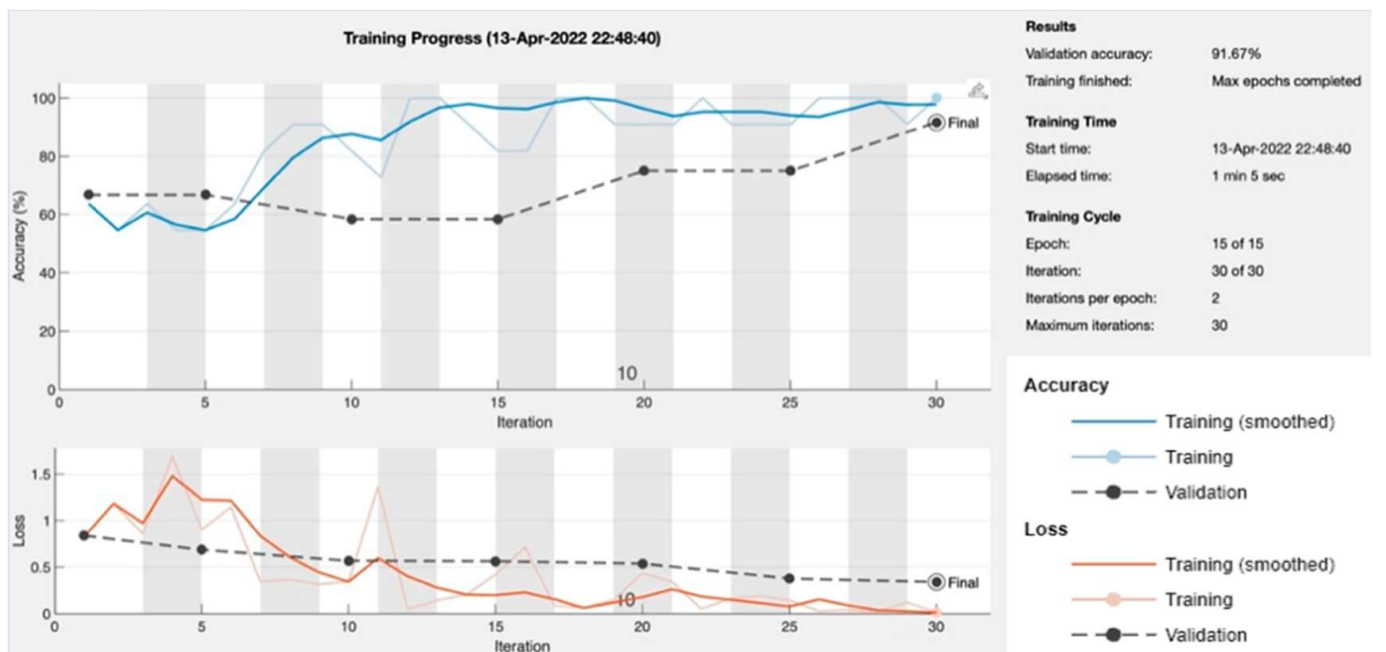


Figure 9. The training performance of AlexNet.

As shown in Figure 10, we used a total of 18 epochs with two iterations for each epoch to properly train and validate the data for ShuffleNet. After 36 iterations, we achieved a prediction performance of 100%, which is optimal accuracy and suggests a well-trained network. Exactly 7 min and 18 s were needed for the training process, far slower than the

time required for the three previous networks. The validation frequency was also carried out using a five-iteration process to prevent data overfitting.

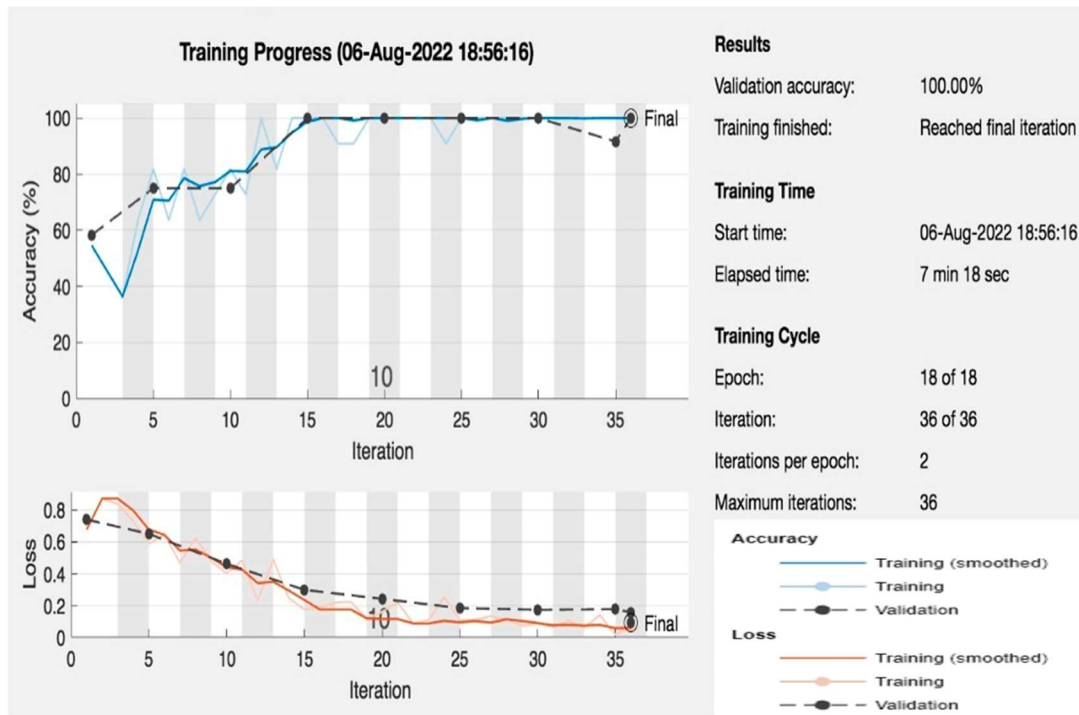


Figure 10. The training performance of ShuffleNet.

As shown in Figure 11 we used a total of 20 epochs with five iterations for each epoch to properly train and validate the data for ResNet50. After 40 iterations, we achieved a predictive performance of 100%, which is optimal accuracy and suggests a well-trained network. Exactly 2 min and 33 s were needed for the training process, a moderate time compared to the other four networks. The validation frequency was also carried out using a five-iteration process.

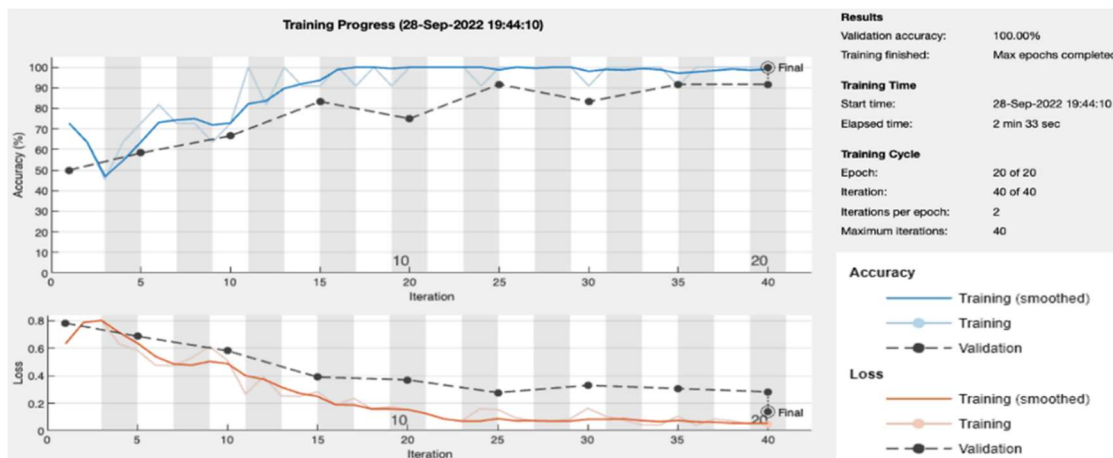


Figure 11. The training performance of ResNet50.

As demonstrated in Figures 7–11, we employed a single CPU system. We used the same initial learning rate of 0.0001 for all of five networks, whereas the maximum iterations and number of epochs were varied for each model. Because SqueezeNet had the fastest training time, it was found to be one of the best networks in terms of validation accuracy.

While ResNet50 also achieved 100% validation accuracy, SqueezeNet was faster than ResNet50. AlexNet and GoogleNet both had the same validation accuracy of 91.67% with a higher training time than SqueezeNet, taking 3 min and 36 s for GoogleNet and 1 min and 5 s for AlexNet. Although ShuffleNet showed perfect validation accuracy, it took a long training time. Table 1 details the training and validation performance of the five networks.

Table 1. Convolution neural network training results.

CNN Model	Validation Accuracy	Elapsed Time	No. of Epochs	Max Iterations	Frequency	Learning Rate	Hardware Resources
Google Net	91.67%	3 min 36 s	60	120	5 iterations	0.0001	Single CPU
Squeeze Net	100%	26 s	20	40	5 iterations	0.0001	Single CPU
Alex Net	91.67%	1 min and 5 s	15	30	5 iterations	0.0001	Single CPU
Shuffle Net	100%	7 min and 18 s	18	36	5 iterations	0.0001	Single CPU
ResNet50	100%	2 min and 33 s	20	40	5 iterations	0.0001	Single CPU

4.2. Model Testing and Evaluation

Our testing dataset consisted of 100 unseen images with 50 drowning instances and 50 normal swimming instances. This testing dataset was a completely separate set which was not used in the training and validation process. Figure 12 presents samples of the testing data. This dataset was used to test the five CNN models, and the testing results are summarized in the confusion matrices presented in Tables 2–6. Samples of the classification results of the five CNN models along with their confidence level are illustrated in Table 7. On the basis of their confusion matrices, the five CNN models were evaluated and compared through six performance measures: accuracy (Ac), recall (R), precision (Pr), specificity (Sp), F1, and MCC. The performance metrics of these models are summarized in Table 8.

Table 2. AlexNet confusion matrix.

		Actual	
		Swimming (–)	Drowning (+)
Predicted	Swimming (–)	50 (TN)	1 (FN)
	Drowning (+)	0 (FP)	49 (TP)

Table 3. GoogleNet confusion matrix.

		Actual	
		Swimming (–)	Drowning (+)
Predicted	Swimming (–)	47 (TN)	2 (FN)
	Drowning (+)	3 (FP)	48 (TP)

Table 4. ShuffleNet confusion matrix.

		Actual	
		Swimming (–)	Drowning (+)
Predicted	Swimming (–)	42 (TN)	11 (FN)
	Drowning (+)	8 (FP)	39 (TP)



(a)



(b)

Figure 12. Samples of testing images: (a) drowning; (b) swimming.

Table 5. SqueezeNet confusion matrix.

		Actual	
		Swimming (−)	Drowning (+)
Predicted	Swimming (−)	49 (TN)	2 (FN)
	Drowning (+)	1 (FP)	48 (TP)

Table 6. ResNet50 confusion matrix.

		Actual	
		Swimming (−)	Drowning (+)
Predicted	Swimming (−)	50 (TN)	0 (FN)
	Drowning (+)	0 (FP)	50 (TP)

Table 7. Samples of testing results of the five CNN models along with their confidence levels.







Testing Samples	CNN Model	Prediction and Confidence Level
	GoogleNet	Swimming 89.4%
	SqueezeNet	Swimming 92.4%
	AlexNet	Swimming 98.8%
	ShuffleNet	Swimming 77.4%
	ResNet50	Swimming 100%
	GoogleNet	Swimming 80.5%
	SqueezeNet	Swimming 94.5%
	AlexNet	Swimming 97.6%
	ShuffleNet	Swimming 75.3%
	ResNet50	Swimming 99.9%
	GoogleNet	Swimming 88.2%
	SqueezeNet	Swimming 90.3%
	AlexNet	Swimming 98.5%
	ShuffleNet	Swimming 73.9%
	ResNet50	Swimming 98%
	GoogleNet	Drowning 98.3%
	SqueezeNet	Drowning 98.7%
	AlexNet	Drowning 99.6%
	ShuffleNet	Drowning 95%
	ResNet50	Drowning 100%
	GoogleNet	Drowning 96.1%
	SqueezeNet	Drowning 96.4%
	AlexNet	Drowning 98.2%
	ShuffleNet	Drowning 92.7%
	ResNet50	Drowning 99.99%
	GoogleNet	Drowning 95.9%
	SqueezeNet	Drowning 97.8%
	AlexNet	Drowning 99.1%
	ShuffleNet	Drowning 93.8%
	ResNet50	Drowning 99.99%

Table 8. Testing results of the five CNN models.

CNN Model	Ac	R	Sp	Pr	F1	MCC
SqueezeNet	97.00%	96.00%	98.00%	97.95%	96.96%	93.97%
GoogleNet	95.00%	96.00%	94.00%	94.12%	95.04%	90.05%
AlexNet	99.00%	98.00%	100.00%	100.00%	98.98%	98.01%
ShuffleNet	81.00%	78.00%	84.00%	82.97%	80.408%	62.11%
ResNet50	100.00%	100.00%	100.00%	100.00%	100.00%	100.00%

The five CNN models were able to correctly distinguish the swimmer’s predicament with a very high confidence level. ResNet50 and AlexNet showed the best prediction performance in terms of all six evaluation metrics. The next best performer was SqueezeNet,

while the ShuffleNet model showed the lowest classification performance among the five CNN models.

To additionally minimize false positive and false negative rates, it would be interesting to combine more than one CNN model, i.e., three or five models, into one system, where the final decision of the system is based on a voting criterion of the individual model outputs.

For further evaluation of our proposed approach, we compared it with other current approaches. The accuracy, machine learning technique, and dataset size of the various systems currently in use are compared in Table 9 and Figure 13. Although it was trained on a smaller amount of data, our proposed approach outperformed other existing approaches in terms of accuracy, as illustrated in Table 9 and Figure 13.

Table 9. Comparison of drowning detection approaches.

Approach	Dataset Size	Training:Testing Data Ratio	Technique	Accuracy
Chan et al. (2020) [12]	4667 images	75%:25%	AlexNet	85.0%
Handalage et al. (2021) [13]	5000 images	90%:10%	YOLO	85.6%
Hasan et al. (2021) [15]	91 videos (46,739 video frames)	90%:10%	MobileNet	96.7%
Our proposed approach	200 images	50%:50%	ResNet50	100.0%

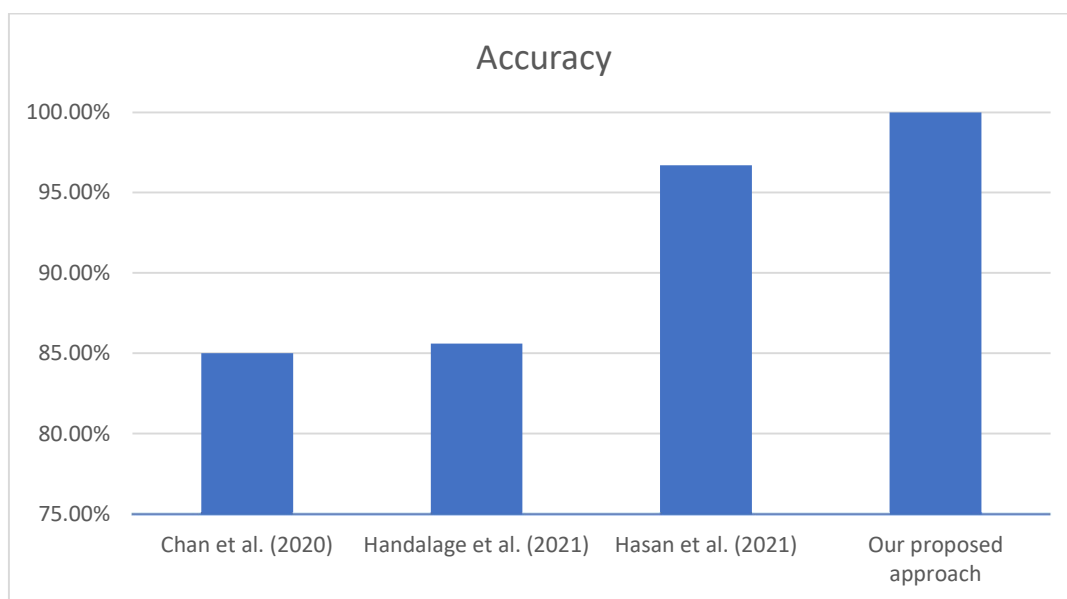


Figure 13. Drowning detection accuracy of existing approaches [12,13,15].

5. Conclusions

This paper presented a deep learning-based approach for early drowning detection. We examined five pretrained convolutional neural networks and trained them on our data. SqueezeNet, GoogleNet, AlexNet, ShuffleNet and ResNet50 as the five networks achieved prediction accuracies of 97%, 95%, 99%, 81%, and 100%, respectively. The best model among them was ResNet50, since it achieved the highest validation and testing accuracy. When compared to other techniques, the system performed exceptionally well in terms of prediction accuracy and training time. Experimental results proved that the proposed models could successfully detect drowning cases within swimming pool environments with very high confidence levels.

The suggested method can be implemented in a variety of pools and settings, including schools, gyms, hotels, and villas. This method can be installed and combined with an alarm system, or it can be integrated with an automated drowning rescue system.

More pretrained CNN models and more drowning/swimming image data can be examined to expand on this research. It would be fascinating to test these models in various swimming conditions with varying lighting and settings. To further minimize false positive and false negative rates, it will be interesting to implement more than one CNN model, i.e., three or five models in one system, with the final decision of the system based on a voting criterion of the model outputs.

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

Funding: This research was funded by the Higher Colleges of Technology (HCT) grant number SURF-243043. Its contents are solely the responsibility of the authors and do not necessarily represent the official views of HCT.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study is collected through the Google search engine and is available upon request.

Acknowledgments: The authors would like to thank Maree Starck for taking the time to review the article for English language.

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

References

1. WHO. *Global Report on Drowning: Preventing a Leading Killer*; World Health Organization: Geneva, Switzerland, 2014.
2. World Health Organization. World Drowning Prevention Day. 2022. Available online: <https://www.who.int/campaigns/world-drowning-prevention-day/2022> (accessed on 20 April 2022).
3. Mohtasham-Amiri, Z. Traumatic injuries in drowning. *J. Inj. Violence Res.* **2022**, *14*, 6, PMID: PMC9115828.
4. Zaara, M.; Belhaj, A.; Naceur, Y.; Makni, C.; Gharbaoui, M.; Bellali, M.; Zhioua, M.; Allouche, M. Patterns of unintentional fatal drowning among children in North Tunisia: A 10-year study. *Rev. D'épidémiologie St. Publique* **2022**, *70*, 31–37. [CrossRef] [PubMed]
5. Krizhevsky, A.; Sutskever, I.; Hinton, G.E.; Krizhevsky, A.; Sutskever, I.; Hinton, G.E. Imagenet Classification with Deep Convolutional Neural. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
6. Szegedy, C.; Liu, W.; Jia, P.; Sermanet, S.; Reed, D.; Anguelov, D.; Erhan, V.V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015.
7. Iandola, F.N.; Han, S.; Moskewicz, M.W.; Ashraf, K.; Dally, W.J.; Keutzer, K. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size. *arXiv* **2016**, arXiv:1602.07360.
8. Zhang, X.; Zhou, X.; Lin, M.; Sun, J. ShuffleNet: An extremely efficient convolutional neural network for mobile devices. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2018.
9. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep residual learning for image recognition. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–30 June 2016.
10. Alotaibi, A. Automated and Intelligent System for Monitoring Swimming Pool Safety Based on the IoT and Transfer Learning. *Electronics* **2020**, *9*, 2082. [CrossRef]
11. Li, D.; Yu, L.; Jin, W.; Zhang, R.; Feng, J.; Fu, N. An Improved Detection Method of Human Target at Sea Based on Yolov3. In Proceedings of the IEEE International Conference on Consumer Electronics and Computer Engineering, Guangzhou, China, 15–17 January 2021.
12. Chan, Y.-T.; Hou, T.-W.; Huang, Y.-L.; Lan, W.-H.; Wang, P.-C.; Lai, C.-T. Implementation of deep-learning-based edge computing for preventing drowning. In Proceedings of the International Conference on Industrial Application Engineering, Taiwan, China, 26–30 March 2020.
13. Handalage, U.; Nikapotha, N.; Subasinghe, C.; Prasanga, T.; Thilakarathna, T.; Kasthurirathna, D. Computer Vision Enabled Drowning Detection System. In Proceedings of the 3rd International Conference on Advancements in Computing (ICAC), Colombo, Sri Lanka, 9–11 December 2021.
14. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You only look once: Unified, real-time object detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 26 June–1 July 2016.

15. Hasan, S.; Joy, J.; Ahsan, F.; Khambaty, H.; Agarwal, M.; Mounsef, J. A Water Behavior Dataset for an Image-Based Drowning Solution. In Proceedings of the 2021 IEEE Green Energy and Smart Systems Conference (IGESSC), Long Beach, CA, USA, 1–2 November 2021.
16. Simonyan, K.; Zisserman, A. Very deep convolutional networks for large-scale image recognition. *arXiv* **2014**, arXiv:1409.1556.
17. Howard, A.G.; Zhu, M.; Chen, B.; Kalenichenko, D.; Wang, W.; Weyand, T.; Andreetto, M.; Adam, H. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv* **2017**, arXiv:1704.04861.
18. Shatnawi, M.; Abdallah, S. Improving handwritten arabic character recognition by modeling human handwriting distortions. *ACM Trans. Asian Low-Resour. Lang. Inf. Process. (TALLIP)* **2016**, *15*, 1–12. [CrossRef]
19. Gandhi, A. Data Augmentation | How to use Deep Learning when you have Limited Data—Part 2. 2018. Available online: <https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/> (accessed on 4 October 2022).
20. MathWorks, “augmentedImageDatastore”. 2022. Available online: https://www.mathworks.com/help/deeplearning/ref/augmentedimagedatastore.html#mw_2ca4481f-3372-415c-803c-e9f30883e93f (accessed on 10 December 2022).
21. Ko, B.C. A brief review of facial emotion recognition based on visual information. *Sensors* **2018**, *18*, 401. [CrossRef] [PubMed]
22. Khan, A.R. Facial Emotion Recognition Using Conventional Machine Learning and Deep Learning Methods: Current Achievements, Analysis and Remaining Challenges. *Information* **2022**, *13*, 268. [CrossRef]
23. Marcelino, P. Transfer Learning from Pre-Trained Models. *Towards Data Science*. 2018. Available online: <https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751> (accessed on 3 September 2022).
24. Ayyar, T.M. A Practical Experiment for Comparing LeNet, AlexNet, VGG and ResNet Models with Their Advantages and Disadvantages. 2020. Available online: <https://tejasmohanayyar.medium.com/a-practical-experiment-for-comparing-lenet-alexnet-vgg-and-resnet-models-with-their-advantages-d932fb7c7d17/> (accessed on 3 October 2022).
25. Khvostikov, A.; Aderghal, K.; Benois-Pineau, J.; Krylov, A.; Catheline, G. 3D CNN-based classification using sMRI and MD-DTI images for Alzheimer disease studies. *arXiv* **2018**, arXiv:1801.05968.
26. Tsang, S.-H. Review: AlexNet, CaffeNet—Winner of ILSVRC 2012 (Image Classification). *A Medium Corp.* 2018. Available online: <https://medium.com/coinmonks/paper-review-of-alexnet-caffenet-winner-in-ilsvrc-2012-image-classification-b93598314160> (accessed on 10 May 2022).
27. Guo, Z.; Chen, Q.; Wu, G.X.Y.; Shibasaki, R.; Shao, X. Village building identification based on ensemble convolutional neural networks. *Sensors* **2017**, *17*, 2487. [CrossRef] [PubMed]
28. Alake, R. Deep Learning: GoogLeNet Explained. 2020. Available online: <https://towardsdatascience.com/deep-learning-googlenet-explained-de8861c82765> (accessed on 4 June 2022).
29. Kurama, V. A Review of Popular Deep Learning Architectures: AlexNet, VGG16, and GoogLeNet. 2020. Available online: <https://blog.paperspace.com/popular-deep-learning-architectures-alexnet-vgg-googlenet/> (accessed on 12 June 2022).
30. Pal, J.B.; Agarwal, S. Real Time Object Detection Can be Embedded on Low Powered Devices. *Int. J. Comput. Sci. Eng.* **2019**, *7*, 417–421. [CrossRef]
31. Mukherjee, S. The Annotated ResNet-50. *Towards Data Science*. 2022. Available online: <https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758> (accessed on 12 June 2022).
32. Shatnawi, M. Review of Recent Protein-Protein Interaction Techniques. In *Emerging Trends in Computational Biology, Bioinformatics, and Systems Biology*; Morgan Kaufmann: Burlington, MA, USA, 2015.
33. Haq, A.U.; Li, J.P.; Khan, J.; Memon, M.H.; Nazir, S.; Ahmad, S.; Khan, G.A.; Ali, A. Intelligent machine learning approach for effective recognition of diabetes in E-healthcare using clinical data. *Sensors* **2020**, *20*, 2649. [CrossRef] [PubMed]
34. Chicco, D.; Jurman, G. The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genom.* **2020**, *21*, 6. [CrossRef] [PubMed]
35. Chicco, D.; Starovoitov, V.; Jurman, G. The benefits of the Matthews correlation coefficient (MCC) over the diagnostic odds ratio (DOR) in binary classification assessment. *IEEE Access* **2021**, *9*, 47112–47124. [CrossRef]
36. Boughorbel, S.; Jarray, F.; El-Anbari, M. Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PLoS ONE* **2017**, *12*, e0177678. [CrossRef] [PubMed]

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