applied sciences

MDPI

Article

U2-Net: A Very-Deep Convolutional Neural Network for Detecting Distracted Drivers

Nawaf O. Alsrehin [1,*,†], Mohit Gupta [1], Izzat Alsmadi [2] and Saif Addeen Alrababah [3]

[1] Computer Sciences Department, University of Wisconsin-Madison, Madison, WI 53706, USA; mohitg@cs.wisc.edu
[2] Department of Computing and Cyber Security, Texas A&M University-San Antonio, San Antonio, TX 78224, USA; izzat.alsmadi@tamusa.edu
[3] Faculty of Information Technology, Al Albayt University, Al-Mafraq 25113, Jordan; saif_r@aabu.edu.jo
* Correspondence: alsrehin@wisc.edu or n_alsrehin@yu.edu.jo
† Current address: Computer Science Department, Yarmouk University, Irbid 21163, Jordan.

Abstract: In recent years, the number of deaths and injuries resulting from traffic accidents has been increasing dramatically all over the world due to distracted drivers. Thus, a key element in developing intelligent vehicles and safe roads is monitoring driver behaviors. In this paper, we modify and extend the U-net convolutional neural network so that it provides deep layers to represent image features and yields more precise classification results. It is the basis of a very deep convolution neural network, called U2-net, to detect distracted drivers. The U2-net model has two paths (contracting and expanding) in addition to a fully-connected dense layer. The contracting path is used to extract the context around the objects to provide better object representation while the symmetric expanding path enables precise localization. The motivation behind this model is that it provides precise object features to provide a better object representation and classification. We used two public datasets: MI-AUC and State Farm, to evaluate the U2 model in detecting distracted driving. The accuracy of U2-net on MI-AUC and State Farm is 98.34 % and 99.64%, respectively. These evaluation results show higher accuracy than achieved by many other state-of-the-art methods.

Keywords: convolutional neural networks; deep learning; distracted drivers; object detection

Citation: Alsrehin, N.O.; Gupta, M.; Alsmadi, I.; Alrababah, S.A. U2-Net: A Very-Deep Convolutional Neural Network for Detecting Distracted Drivers. *Appl. Sci.* **2023**, *13*, 11898. https://doi.org/10.3390/app132111898

Academic Editors: Junhui Huang, Qi Xue and Zhao Wang

Received: 17 September 2023
Revised: 18 October 2023
Accepted: 20 October 2023
Published: 31 October 2023

1. Introduction and Background

Driving is a common activity for many people, it provides social interactions with other people on the road, facilitates human independence, and offers a significant chance for exploring the world [1]. The Department of Motor Vehicles reported that there are currently around 228.2 million driving licenses in the United States and the Road, and Highway construction in the United States (U.S.) has been increasing by 0.7% from 2021 [2]. Moreover, there are over 284 million vehicles operating on roads throughout the United States. Therefore, making driving as safe as possible is an important issue in every day life.

Despite the safety improvements in road construction and vehicle manufacturing, the total number of vehicle crashes is still increasing. The National Highway Traffic Safety Administration (NHTSA) reported that 90% of road accidents in the U.S. are due to human errors and 20% of traffic accidents are caused by distracted drivers. In 2020, 3142 people were killed by distracted driving in the U.S [3]. Using a mobile phone while driving has become one of the biggest factors responsible for accidental injuries and deaths in the U.S.

Distracted driving is any activity that diverts attention from driving including eating, drinking, talking to passengers, texting, adjusting the radio, or using navigation [3]. Distracted driving can cause serious safety concerns for drivers as well as passengers. Among several activities taking the driver's attention off the road, it is observed that a high-level distraction is caused by using mobile phones. The National Safety Council reported that texting while driving leads to 1.6 million crashes and nearly 390,000 injuries [1]. If driver

distraction is detected in real-time, and if an early warning is given, this can avoid traffic accidents. The focus of this study is to automatically detect driver's distraction caused by using mobile phones. According to a report by the National Safety Council in U.S., using mobile phones while driving is considered a high-level distraction that leads to millions of crashes and thousands of injuries. A driver's behaviors of "adjusting radio" or "talking with passengers" are considered as distraction behaviors in two benchmark datasets involved in this research. Currently, Driver Assistance Technologies (DAT) hold the potential to reduce traffic crashes and save thousands of lives each year [3]. DAT are groups of electronic technologies that assist drivers in driving and parking functions such as avoiding collisions, pedestrian detection, and lane change warning. DAT use various sensors, computer vision, and machine learning algorithms to monitor the vehicle's surroundings and make driving tasks easier and safer.

There have been many efforts to detect distracted drivers due to use of mobile phones while driving. Research shows that drivers on phone calls were not able to perceive risks. In fact, people who drive on phone calls increase the risk of a crash by about 4 and 6 times [2]. Detecting drivers using mobile phones can be done in several ways, such as: using surveillance cameras, smart technologies, and mobile applications. Using image and video processing algorithms to detect the usage of mobile phones is considered the most reliable and widespread method due to several factors, such as: the accuracy of their results, concrete evidence when required, and it is easy to keep the data generated for long periods of time.

Traffic management systems, local governments, and traffic enforcement systems currently use cameras that are already installed and distributed all around the roadways for security, monitoring, and object detection purposes. Object detection deals with detecting instances of semantic objects of a certain class in digital images and videos. Object detection is an active and well-researched domain that includes face detection, pedestrian detection, distracted driver detection, vehicle detection, and more. Artificial Intelligence (AI) and Machine Learning (ML) applications are increasing daily. Deep Neural Networks (DNN) are an important topic in the field of artificial intelligence. Image and video processing uses AI and ML in several applications, such as in object detection. Using AI, ML, and DNN algorithms and techniques for object detection increases the accuracy of the detection and expands its applications. Deep learning can play a significant role by detecting the distracted driver activity and notifying the driver to stop while these activities occur to prevent accidents.

While U-Net may not be a suitable choice for object tracking on its own, it is worth mentioning that features extracted by U-Net architecture can be valuable for object detection tasks [4]. Object tracking requires algorithms and architectures that are specifically designed to handle the temporal continuity and identity maintenance of objects across video frames. The U2-net architecture does not handle the temporal information and object identity across video frames. However, considering the temporal continuity of objects between frames will be considered when dealing with videos, and represents a future work that is outside the scope of this research. In this research, we develop a model called U2-net that automatically detects distracted drivers, especially those who are using mobile phones. The U2-net is a modified version of the original U-net architecture. U-Net is a convolutional neural network (CNN) architecture that consists of an encoding path and a corresponding decoding path, with skip connections that bridge the encoding and decoding paths at multiple resolutions. The encoding path is a series of convolutional and pooling layers that gradually reduce the spatial resolution of the input image, while the decoding path consists of transposed convolutional layers that progressively up-sample the feature maps to the original resolution. The skip connections combine the low-level feature maps from the encoding path with the high-level feature maps from the decoding path to capture both local and global information in the image. U-Net is a 2D CNN architecture that is widely used for image segmentation [4,5].

We modify the original U-net architecture by adding four convolution layers, two max pooling layers, and dropout layers to the contracting path. additionally, in the expanding path, we add four transpose convolution layers, two max pooling layers, and dropout layers. These new layers add more spatial information and context to the classifier and avoid model overfitting. Moreover, we use full-color images to provide more information to the classifier and educe the size of each input image to reduce the training time and minimize the model complexity. The dense layer is used to predict the output result that is formed by a weighted average function of its input. The U2-net model provides deep layers to represent image features and yields more precise classification results. The motivation behind this model is that it captures the context around the objects in order to provide a better object representation and classification.

The U2-net uses input images collected from cameras that capture frontal views from inside the vehicle. Many existing methods have some limitations, such as being locally integrated inside the vehicle, and therefore cannot prohibit distracted drivers. Additionally, their accuracy drops if the driver changes his behavior, Moreover, they do not provide concrete evidence. The U2-net model addresses some of these limitations by providing a dynamic and adaptive system that provides efficient results regardless of the driver behavior. It achieves 2.75% higher accuracy results than the D-HCNN [6] on the MI-AUC dataset and 1.52% higher accuracy results than the MobileNetV2 [7] on the State Farm dataset. The U2-net model can also help increase the drivers' awareness of their driving habits and associated risks, promote safe driving practices, and help vehicle manufacturers improve the safety levels in vehicles and minimize the driver distraction. In addition, it can be integrated with smart surveillance cameras that are available on roadways and controlled by the Central Traffic Department. Additionally, the U2-net model is responsible for saving the captured images/video for the distracted driver, especially those who use their phone while driving, and send captured images to the main database for manual confirmation to take the correct action.

The main contribution of our work can be summarized as follows:

1. We develop a new model called U2-net that automatically detects distracted drivers, especially those who are using mobile phones;
2. We modify the original U-net model by adding more layers to both contracting and expanding paths, which can capture more intricate features and patterns in the data. Additionally, we add a flattened layer along a dense layer, which helps in the detection and classification process;
3. We update the regularization technique, optimization technique, and the loss function to improve the performance and avoid model overfitting;
4. We apply the U2-net into the image classification domain and evaluate it using two datasets: MI-AUC and State Farm datasets;
5. The evaluation results show that the U2-net model achieves higher accuracy, recall, precision, IoU, and F-score than what is achieved by many other state-of-the-art models.

2. Related Work

There are active efforts in developing methods for detecting distracted drivers. Because of the impact of distraction while driving on public safety and property, several governments and countries have enacted regulations that prohibit any cause of distraction while driving, especially mobile phone usage. These regulations have created a need for a smart system to detect mobile phone usage while driving for law enforcement. Currently, the driver's use of mobile phones is mainly supervised by law enforcement officials through direct observation. It is time-consuming, inefficient, and almost impossible to cover all roadways. So, the challenge is to develop a smart system that automatically detects the usage of mobile phones while driving. There are different ways to solve such problems. First, we review detection methods that are different from visual-based methods. Then, we review visual-based methods that use AI and machine learning algorithms.

Detection using Sensors and Equipment: There are different ways to solve such problems. For example, one system uses sensors and equipment inside the vehicle, such as Hands on the Steering Wheel [8], which block any notification on a smartphone for some period while the driver is conducting the vehicle. However, this system cannot avoid misuse. Another approach of Yang et al. [9] used software running on the phone for capturing and processing high frequency sound signals sent by the car sound equipment. The sound signals are used to measure the position of the mobile phone and when the driver is talking on it [10]. However, it depends on the mobile phone brand and the software must be continually enabled by the driver. A general and efficient way is to develop an image-based system that considers all different types of vehicles, gestures, positions, and mobile sizes. In addition, it should handle different weather conditions. All these situations make detection hard. The positions of cameras also affect the detection accuracy, which is either inside or outside the vehicle. Vehicle manufactures equip vehicles with inside cameras to enhance the safety level while driving, while the outside cameras are generally installed to detect illegal behaviors.

Detection using Deep Learning: Hesham Eraaqi et al. [11] proposed a reliable deep leaning-based solution that achieves 90% accuracy. Additionally, the authors presented a genetically weighted ensemble of convolutional neural network for image classification that achieves 84.64% classification accuracy. Celaya-Padilla et al. [12] proposed a novel approach to detect distracted drivers using mobile phones while driving. The authors used a ceiling-mounted wide angle camera coupled to a deep learning-convolutional neural network (CNN) to detect distracted drivers. The authors used Inception V3 deep neural network, which was trained to detect "texting and driving" subjects. The final CNN was trained and validated on a dataset of 85,401 images, achieving an area under the curve (AUC) of 0.891 in the training set, an AUC of 0.86 on a blind test, and a sensitivity value of 0.97 on the blind test.

Uzzol Hossain et al. [7] developed a CNN-based method to detect distracted drivers and identify the cause of distractions such as talking, sleeping, or eating by means of face and hand localization. Four architectures, namely CNN, VGG-16, ResNet50, and MobileNetV2 have been adopted for transfer learning. The proposed model was trained with thousands of images from the state farm dataset containing 10 different postures or conditions of a distracted driver and analyzed the results using various performance metrics. The performance results showed that the pre-trained ResNet50 and Mo-bileNetV2 provide the best classification accuracy of 94.5% and 98.12%, respectively.

Detection using Combination Methods: Mohammed S. Majdi et.al. [13] presented an automated supervised learning method called Drive-Net for driver distraction detection. The authors used a combination of a convolutional network (CN) and a random decision forest for classifying images of a driver. The authors compared the performance of the Drive-Net with two other popular machine-learning approaches: Recurrent Neural Network (RNN), and a multi-layer perceptron (MLP). Using about 22,425 acquired images under a controlled environment that are publicly available and manually annotated by an expert, Drive-Net achieves a detection accuracy of 95%.

Binbin Qin et al. [6] proposed a new D-HCNN model based on a decreasing filter size with a smaller number of parameters than that used by models in many other studies. D-HCNN uses HOG feature images, L2 weight regularization, dropout and batch normalization to improve the performance. The authors conducted experimental evaluations on two public datasets: MI-AUC [7] and State Farm [14] with an accuracy of 95.59% and 99.87%, respectively, which represent higher accuracy values than those achieved by many other state-of-the-art methods.

There are several other attempts to develop efficient deep learning frameworks to detect distracted drivers such as EfficientDet [15] and other models, described in Ref. [16]. These attempts have analyzed the static characteristics of images. Other attempts, described in Ref. [17] discover the temporal–spatial characteristics of images. Some of these attempts were done by fusing the attention features extracted from the dynamic optical flow in-

formation and the spatial feature of the single video frame to recognize the distracted driving posture [17]. Others have combined the handcrafted features and fusion features in a hybrid scheme [18].

Wang et al. [19] developed an algorithm that uses input data generated from a camera inside the car. Such a system is not useful in all vehicles because drivers may easily block it while using a mobile phone and one cannot prohibit the driver from doing it.

Another study was done by Artan et al. [20] that intended to get pictures from a front windshield view of the vehicle. First, the driver's face was detected, then Region of Interest (ROI) was identified, and finally mobile phone usage gestures were detected. The detection approach was done by training the system with positive and negative pictures. Positive pictures are with mobile phone usage and negative pictures are without. They proposed two classification architectures, used full and half face images, and the performance in terms of accuracy, specificity, and sensitivity was compared. The evaluation results showed that the performances of two architectures are similar, above 86% for the mobile phone usage detection task.

A neural network-based application was developed to detect mobile phone usage [21]. Sample positive (with phone) and negative pictures (without phone) were used for training and testing the cascade object detector on MATLAB. Initial evaluation results showed that the quality of the captured image and the number of images used for training play a major role in the detection accuracy, which was 75%.

Berri et al. [22] used Support Vector Machine (SVM) with Polynomial kernel classification to develop an algorithm for extracting characteristics allowing mobile phone identification while driving a vehicle. They used sets of images with 100 positive and other 100 negative images containing frontal images of the driver. The evaluation results showed a success rate of 91.57% for the vision system.

For a 3D vision system, Berri and Osorio [23] proposed a solution that uses Short-Term (ST) and Long-Term (LT) pattern recognition subsystems to analyze the positions of the driver's hands. The system has three levels of alarms: no alarm, lowest alarm, and highest alarm. ST detects between no alarm or some level alarm. LT is responsible for determining the risk level, either low or high. The classifiers are based on ML and Artificial Neural Networks (ANN). Furthermore, the values set to adjust input features, neuron activation functions, and network topology/training parameters were optimized and selected using a Genetic Algorithm. The experiments achieved 95% accuracy as the best system performance results.

Aljohani [24] proposed a model called "DenseNet-GA" that combines artificial deep learning and machine learning models with genetic algorithm for actions detection of drivers from input images. Two dense layers, K nearest neighbor, random forest, support vector machine, and extreme boost algorithms have been used as classifiers. The DenseNet-GA model was developed with the use of a state farm dataset that contains information of one safe driving class and nine dangerous behaviors. Experimental results show that the classification accuracy of the DenseNet-GA is 99.80% when using the state farm dataset. However, it is highly recommend that the proposed model is evaluated using another dataset.

Detection using U-net Model: One of the most efficient CNN approaches is U-Net, which is mainly used for medical image segmentation. U-Net contains a shrinking path for deriving information and a corresponding expansion path for restoration and has high accuracy for image segmentation [25]. Many researchers have investigated U-Net because of its popularity and efficiency in image segmentation in several domains such as cytology [5], geology [26], and microorganisms [27]. Although the original U-Net has a strong versatility in achieving good results in image segmentation, many variant network structures have been proposed to improve the segmentation effect [28]. To the best of our knowledge, no research has previously investigated the U-Net to address the problem of detecting and classifying the driver's state of distraction while driving, which might be used to alert the driver. This paper introduces an improved version of U-Net model that

uses deep feature maps, dense layers, more convolution, max pooling, and dropout layers to add more spatial information and context to the classifier, which enhances the accuracy of the classification results.

There is a similar attempt that extends the U-Net architecture for volumetric medical image segmentation tasks called V-Net, which is a 3D CNN architecture that uses a similar encoding-decoding structure to U-Net but incorporates 3D convolutional layers instead of 2D convolutional layers. V-Net also uses residual connections to improve the gradient flow and speed up the convergence during training [29]. Table 1 summarizes the related work in terms of the proposed method, architecture, dataset, and resulted accuracy.

Table 1. Comparison between the related work in terms of architecture, dataset, accuracy, limitation, and research gap.

Reference	Method	Architecture	Dataset	Accuracy
[6]	HOG, L2, dropout, and batch normalization	HOG + CNN	MI-AUC and State-Farm	95.59% 99.87%
[7]	CNN, VGG-16, ResNet50, MobileNetV2	Transfer Learning	State-Farm	97.45%, 94.01%, 94.28%, 98.12%, respectively
[11]	Deep Learning	Ensemble CNN	MI-AUC	90%
[12]	Deep CNN	Inception V3	MI-AUC	89.1%
[13]	Drive-Net	CNN with Random Forest	State-Farm	95%
[17]	TSD-DLN and C-AOG	Deep Learning	MI-AUC	88.3%
[30]	Deep CNN	VGG16 and VGG19	State-Farm	98.98%
[24]	DenseNet-GA	Genetic Algorithm with Deep NN	State-Farm	99.80%
[31]	CAT-CapsNet	Capsule network with attention module and convolution filters	State-Farm and MI-AUC	99.88%, 96.78%

3. The U2-Net Architecture

In this paper, we adjust and prolong the U-net architecture [5] in a way that yields more precise image classification results. The U-net architecture was initially used to enhance image segmentation. It offers a training technique that greatly hinges on data augmentation and uses the prevailing annotated models well. The characteristics of the U-Net architecture, such as skip connections, fully convolutional design, and efficient training, have made it adaptable to a wide range of computer vision tasks beyond medical imaging. It has been successfully used in applications such as satellite image segmentation, road segmentation in autonomous driving, and more. Generally, the Fully Convolutional Networks (FCNs) enable end-to-end learning for pixel-wise predictions and capture local and global context information through convolutional layers. The U-Net architecture builds on the concept of (FCNs), which has shown promise in image segmentation tasks. The U-shape architecture allows high-level features and fine-grained details. The skip connections help preserve spatial information and enable the model to recover details lost during the down-sampling process and enhance accuracy. U-Net was designed with computational efficiency in mind, making it feasible for real-time or near-real-time applications. This efficiency is crucial in detecting distracted drivers, where quick results can have a significant impact on people who are using the road. In summary, the U-net architecture along with its innovative design and adaptability have made it a widely adopted and influential model in the field of image classification.

There are different neural network structures that have been used to detect distracted drivers such as VGGNet [30], ResNet [32], and EfficientNet [33]. However, these structures do not preserve image spatial information nor concatenate image features, which are critical in our task. The strengths of U-Net lie in its suitability for recovering fine-grained details and spatial information [34]. The novelty of the U2-net model are its structure, ability to adapt to different domains, and its ability to customize the loss function. Adding the color information to the model along with creating new different convolutional layers and aggregating them with dense and flattened layers allows the U2-net model to capture distinct visual cues of distracted driving and enhance the classification results. U2-net increases the depth of the whole U-net architecture without significantly increasing the computational cost because of the pooling operations.

The original U-net architecture is made up of two paths: expanding and contracting path. The encoder (contracting path) captures the image's context, whereas the uniform, increasing path (the decoder) allows for exact location [5]. The contracting route consists of

max pooling and convolution layers, whereas the expanding path consists of transpose convolution operation and fully connected layers. The contracting path catches the information around the objects to offer a more accurate representation of the object, which is the reason behind this design. The U-Net design serves as the foundation for our U2-net. Figure 1 shows the architecture design of the U2-net model, which is created to automatically identify distracted drivers using input image.

In the U2-net model, we modify the unique U-net architecture by adding four convolution layers and two max pooling coatings to the contracting pathway. In addition, in the escalating path, we add four up-sampling (transpose convolution) layers and two max pooling layers. These new convolution layers add more spatial information and context to the classifier. Additionally, we use full-color images (RGB color channels) instead of gray-scale images, as in the U-net architecture. The transpose convolution layers upsurge the tenacity of the output. To keep sound localization and reduce the time set for training, we decrease the dimension of each input picture, making them 256 × 256 pixels in size and fodder as the input to the proposed architecture.

For each input image, convolutional operations are performed to extract image features, followed by pooling operators to reduce the image dimension. These convolutions and pooling operators are performed for the successive layers to reach the minimum image dimension that is full of features. Then the transpose convolution is accomplished to up-sample the input feature map to an anticipated output feature map. After that, concatenating the layers generated from the pooling operators with layers generated from the up-sampling operators is performed to increase the resolution of the results and to help extract the prevailing features that are essential for operational arrangement and exposure. A significant array of feature systems enable the web to transmit contextual data to more excellent firmness layers during the up-sampling process. The U2-net design, such as the U-net architecture, has both a contracting route (Figure 1: left side) and an expanding path (Figure 1: right side). The narrowing path is measured in the manner of a convolutional system. It comprises two 3 × 3 convolution layers applied repeatedly and one max pooling process. Each convolution employs the same Rectified Linear Unit (ReLU) and buffering. The stride of 2 is used for down-sampling using the identical buffer in the 2 × 2 max pooling procedure.

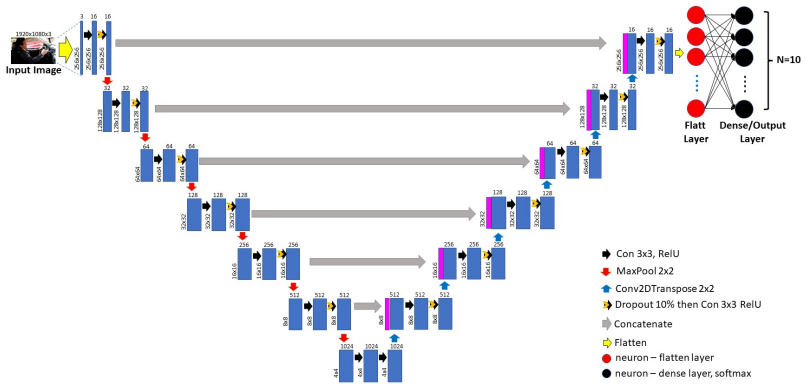

Figure 1. The architecture for the U2-net model, starting from the input image (**left side**) to the output classifier (**right side**). Figure 2 provides more details about these layers.

The U2-net architecture takes a set of images as input to generate a list of probabilities based on the number of classes as an output. Each image from each dataset represents an input to the contracting operation. The contracting operation begins with 16 filters (feature channels) and ends with 1024 filters, while the expansive operation begins with 1024 filters and ends with 16 filters. At each step in the contracting path, we drop 10% to avoid model overfitting and we double the number of feature channels to reach 1024 feature channels.

The feature map's up-sampling (transpose convolutions) is the first stage in the expanding path. This is monitored by a 2 × 2 convolution, which cuts the number of feature channels in half, a concatenation with the corresponding feature map from the contracting path, and two 3 × 3 convolutions, each followed by ReLU activation function with the stride of 1 and with the same padding. In addition, we drop 10% each time to avoid model overfitting. A successive dense layer is used to gather a correct output, enhancing the accuracy. In total, the U2-net architecture has 1 input layer, 32 convolutional layers, 6 max-pooling layers, 6 transpose convolution layers, 6 concatenating operations, 12 dropout layers, and 1 dense layer, as follows and shown in Figure 2.

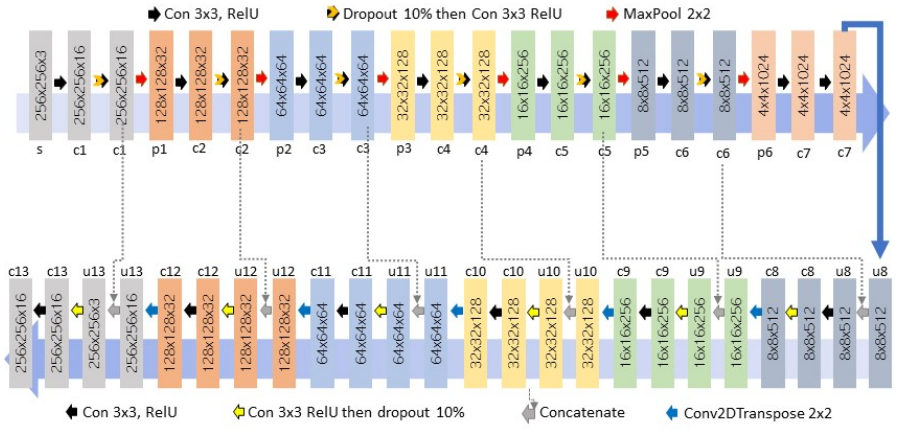

Figure 2. The architecture for the U2-net model, starting from the top-left layer.

1. *Input layer*. Each image from the dataset represents an input to the U2-model. We reprocess each input image by reducing the dimension to 256 × 256 × 3 with RGB color channels using a bi-cubic interpolation algorithm. Resizing plays a major role to train any Machine Learning (ML) or Deep Learning (DL) models.

2. *Convolutional layer*. We use the typical convolutional layer in the U-net architecture generated by applying the convolution operation using a kernel of different sizes, number of filters, stride, padding, and activation function. This layer contains all the vital topographies of an image, such as color and gradient orientation. The input to the convolution layer is a matrix containing image information, while the output is a map full of image features. We used nested convolution operation along with the ReLU as an activation function. It is the most used initiation task in DL replicas that proposes supplementing and strengthening the nonlinearity properties of the neural system. Each image has a set of channels. We start with 16 different channels as an initial step and then duplicate it for each next layer in the encoding path, as described in Figure 1 to reach out to 1024 channels. while in the decoding path, we start with 1024 and reduce this number by dividing it by 2 for each level to reach out to the original number 16. We use the *Conv2D* function provided by *tensorflow.Keras* to perform the convolution step. For any given input x, the ReLU is defined as follows [35,36]:

$$RelU(x) = \begin{cases} x, & \text{if x > 0} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

3. *Max pooling layer*. The down-sampling layer is another name for this layer. It is the layer handled after the convolution layer in the CNN design. The feature maps' height and breadth are reduced while maintaining the same depth by the max pooling layer. To prevent model overfitting, reducing the map's size, the amount of computation required, and the number of factors, is helpful. The down-sampling operation can be performed by using either maximum pooling or average pooling. Maximum pooling extracts the maximum value of the elements within a receptive field, and average pooling calculates

the mean value. We used the maximum pooling as a pooling layer because it preserves the features in the input feature map in a better way in terms of object representation during the down-sampling process [31,37]. We use the *MaxPooling2D* function provided by *tensorflow.Keras* to perform the max pooling step.

4. *Transpose convolution layer*. This layer is also known as the up-sampling or deconvolutional layer. We use this layer in the decoding path to precisely restore and locate features. It represents standard convolutions defined by the padding and stride and is usually carried out for up-sampling to generate an output feature map that has a spatial dimension greater than that of the input feature map. We use the *Conv2DTranspose* function provided by *tensorflow.Keras* to perform the transpose convolution step. For a given size of the input i, kernel k, padding p, and stride s, the size of the output feature map o generated is given by:

$$o = (i - 1)s + k - 2p \tag{2}$$

5. *Dropout layer*. The dropout layer randomly deletes inputs based on the rate (we use 10%) at each update during training time, which helps prevent overfitting. We used the dropout operation after each first convolution when the max pooling, or up-sample operation, is completed. We use the *Dropout* function provided by *tensorflow.Keras* to perform the dropout step.

6. *Dense layer (fully connected layer)*. In any neural network, a dense layer is a layer that is fully connected with its preceding layer. It analyses the features obtained from the fully connected preceding layers and predicts the output result that is formed by a weighted average function of its input. The weighted average is passed through an activation function that generates the output score of that neuron. Similarly, the process is carried out for all neurons of all layers.

For multi-class classification problems, the soft-max function is the most commonly used activation function. The number of dense layers depends on the problem requirement and number of output classes. Each neuron in the last dense layer has a vector that contains all the probabilities of an element belonging to each class as a classification result. The predicting class is the class that has the highest probability value. We used one *Flatten* layer and one *Dense* layer along with the *Soft-max* function [38] provided by *tensorflow.Keras* to represent the output score value as:

$$y_i = \frac{e^{(W_{xi}+B)}}{\sum_{i=1}^{N} e^{(W_{xi}+B)}} \tag{3}$$

where W and B stand for the amount of weight and biases in the preceding layer, respectively, and y_i gives the clustering likelihood for the specific class i. The overall amount of courses is N. In this paper, we use N = 10.

7. *Loss function*. The value disparity between the expected and actual outcomes is the loss function. It evaluates how effectively the neural network simulates the training set of data. The algorithm performs better when the number is lower. Any model's training goal is to reduce the loss value as much as possible. Cross entropy is a popular function of loss for classification algorithms. An optimization function known as the cross-entropy loss function is employed when building a classification model to categorize data by estimating the likelihood that the data will pertain to one class or another. For binary classification, binary cross entropy is frequently used as a loss function, and for multi-class classification, categorical and sparse categorical cross-entropy are commonly used. In the U2-net model, we used the *categorical_crossentropy* categorical cross entropy supported by *TensorFlow.Keras*.

8. *Optimizer*. An optimizer is a program or function that modifies the training rate and weights of the neural network. As a result, it enhances precision and reduces overall waste. The optimizer is utilized from the software. we use *adam* optimizer provided by *TensorFlow* deep learning framework, which provides proven effectiveness and optimized implementations. The adam optimizer changes how the model's parameters should be

modified and adapts the learning rate to allow for efficient input processing when combined with the momentum idea [39]. *TensorFlow* also provides *adam* optimizer under *Keras*.

9. *Concatenation*. The concatenate layers play a crucial role in merging the outputs of different layers into a single layer. This allows the model to learn shared features across tasks, leading to improved performance, efficient training, and better generalization [5]. We used the *concatenate* function supported by *TensorFlow.Keras*.

4. Experimental Results and Analysis

We evaluated and verified the U2-Net using a Google Colab Pro with 32 GB of RAM, an NVIDIA P100 Tensor GPU, and two virtual CPUs. Figure 3 shows the flowchart of the recommended methodology. It shows the development and evaluation processes we follow to reach the results. It shows four general steps: the Datasets step focuses on the dataset we have selected to evaluate the U2-net model; more details will be presented in Section 4.1. The Data Pre-processing step shows our operations to prepare the data as an input to the U2-net mode. It is designated in Section 4.2. Section 4.3 depicts the Model Training step in which we describe how we use the datasets to train the U2-net model along with the model structure, layers, input, and output. The final step is the Model Evaluation step, in which we describe the evaluation metrics we used, and this step is described more in Section 4.4.

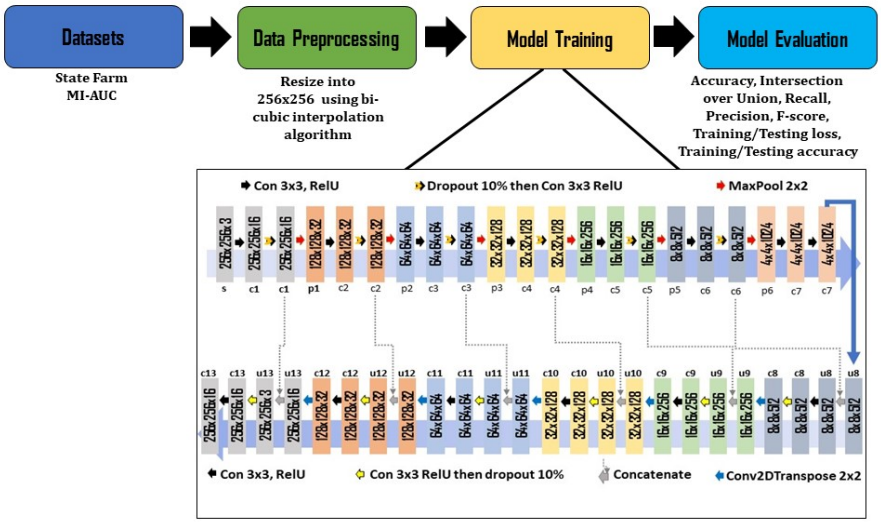

Figure 3. The general structure of the research methodology.

4.1. Datasets

Table 2 shows sample images from the two datasets we have used to evaluate the efficiency of the U2-net model. The Machine Intelligence group generated the first dataset from the American University in Cairo (MI-AUC) [11]. We contacted the authors, who generously provided us with the link to download it. It contains 14,478 images distributed over 10 different class postures of the distracted driver (Co: Safe Driving, C1: Phone Right, C2: Phone Left, C3: Text Right, C4: Text Left, C5: Adjusting Radio, C6: Drinking, C7: Hair or Makeup, C8: Reaching behind, C9: Talking to Passengers). These images were extracted from videos captured by a camera installed on the car's roof on top of the passenger's seat. Each captured image is either 1920 × 1080 pixels or 640 × 480 pixels with RGB color model. The second dataset is the first publicly available dataset generated by State Farm's distracted driver competition on Kaggle [14]. It contains 22,424 images distributed over the same 10 classes. These images were captured by a camera installed on the car dashboard. Each captured image is 640 × 480 pixels with an RGB color model. Table 3 shows the total number of images (32,833) used from both datasets to train and test the U2 model.

Table 2. Sample images from the two datasets used in this study.

Class	Number of Images	Dataset1-MI-AUC [11]		Dataset2–State Farm [14]	
C0	Dataset 1: 2986 Dataset 2: 2489				
C1	Dataset 1: 1256 Dataset 2: 2267				
C2	Dataset 1: 1718 Dataset 2: 2317				
C3	Dataset 1: 1718 Dataset 2: 2346				
C4	Dataset 1: 1124 Database 2: 2326				
C5	Dataset 1: 1123 Database 2: 2312				
C6	Dataset 1: 1076 Database 2: 2325				
C7	Dataset 1: 1044 Database 2: 2002				
C8	Dataset 1: 1034 Database 2: 1911				
C9	Dataset 1: 1797 Database 2: 2129				

Table 3. The number of images used for training and testing.

Dataset	Training Data	Testing Data	Total
State Farm [14]	17,829	4458	22,278
MI-AUC [11]	8444	2111	10,555

4.2. Data Preprocessing

Processing the images before fitting them into the learning model is essential to attain more accurate results. We extract the RGB channels from each image to provide better spatial resolution. The original U-net model deals with only one channel from each input image. Color images contain more information that provides better classification results and high-resolution feature maps [40]. Each image is then resized into 256 × 256 pixels using the bi-cubic interpolation algorithm to facilitate the batch learning process and reduce the computational complexity.

4.3. Model Training

Each dataset is used to train the U2-net design. A total of 80% and 20% of the overall sample amount comprises training and test specimens from each dataset, which are randomly selected. The quantity of images used for model testing and training across the two datasets is displayed in Table 3. Each input image is subjected to a set of filters in the U2-net design, which produces a feature map that lists the existence of any identified features. The U2-Net is designed for detecting distracted drivers without using any pre-trained backbones from image classification. Hence, all of our convolutional layers are initialized and trained from scratch to achieve competitive performance. One significant change we made to the original U-net design was to have the network transmit numerous feature channels to higher-density layers during the increased sampling process. These layers could learn the model and produce a more accurate final result. The U2-net architecture has a fully connected dense layer following the final convolution layer, as shown in Figure 1, which normalizes the data in the feature space and displays the classification results as probabilities, unlike the original U-net architecture, which only utilizes the legitimate part of each convolution. We use the the Softmax activation function as logistic regression layer

to achieve classification, which is based on the principle of regression, and its loss function is a probabilistic model considering global data. It normalizes the data in the feature space and presents the classification results in the form of probabilities. Each neuron in the final dense layer in U2-net model has the value $Y = Y_1, Y_2, Y_3, ..., Y_N$ as a vector, where Y_i is the possibility that the input image Y fits to a class i. The highest probability value is selected as a label of the input image Y. Note that the total number of clusters is N = 10. Adam optimizer is used to train the U2-net network and its hyper parameters are set to default (initial learning rate lr = 1×10^{-3}, betas = (0.9, 0.999), eps = 1×10^{-8}, weight decay = 0). We train the network at a batch size of 64, verbose = 1, and epochs = 50 until the loss converges without using the validation set, which follows the original U-net model. The training loss converges at early stopping (mode = max) and the whole training process takes about 20 h.

For the convolution operation, the kernel size is 3 × 3, number of filters = (16, 32, 64, 128, 256, 512, 1024), stride = 2, padding = same, activation function = 'relu', kernel initializer = 'he_normal'. For the transpose convolution operation, filters = (512, 256, 128, 64, 32, 16), kernel size = 2 × 2, strides = (2,2), padding = same. To minimize the processing complexity, the pooling layer (stride = 2 and pool size of 2 × 2 with no padding) is in charge of lowering the size of the recovered vectors and dimensionality. Additionally, it can locate and retrieve the main features required for a classification or detection method to work effectively.

4.4. Model Evaluation

We used the accuracy metrics provided by *TensorFlow.Keras*. The accuracy represents how often the predicted values match the labels. Figure 4 shows the evaluation results in terms of accuracy and loss percentages for the MI-AUC and State Farm datasets. While the test accuracy shows how well the model matches the testing data, the training accuracy shows how well it fits the training data. The testing loss evaluates the inaccurate prediction of the model on the testing set and shows the disparity between the model and the testing data. The training loss evaluates the inaccuracy of the models, it demonstrates the disparity between the model and the training set. Note that the training set represents 80% of the dataset initially used to train the model. The testing accuracy curve is close enough to the training accuracy curve. In addition, we can notice that the U2-net model achieves high training and testing accuracy values. Further, the U2-net model achieves better results when using the State Farm datasets. Moreover, we notice that the U2-Net model starts to converge at early stages (i.e., when epochs < 5).

Table 4 shows the training accuracy, testing loss, training loss, and testing accuracy for U2-net compared to other architectures using the State Farm datasets. The results in the first four rows are collected from Ref. [7]. The authors in Ref. [7] have implemented and evaluated these architectures using the same datasets that we used—State Farm. They also used the same training and testing ratios (80% for training and 20% for testing). The last row shows the evaluation results of the U2-net on the State Farm datasets. We notice that the U2-net achieves the best results regarding accuracy and loss values. It achieves 100% training accuracy and 99.64% testing accuracy (1.52 higher than MobileNetV2 [7]).

Table 4. Training and testing accuracy for U2-NET compared to other architectures using the State Farm dataset [7].

Model	Training Loss	Training Accuracy	Testing Loss	Testing Accuracy
Simple CNN [7]	0.147	96.09%	0.1348	96.76%
VGG-16 [7]	0.00185	97.68%	0.2424	93.23%
ResNet50 [7]	0.0597	98.62%	0.26795	94.69%
MobileNetV2	0.0035	99.83%	0.2057	98.12%
U2-net (proposed model)	**0.0000081**	**100.0%**	**0.02**	**99.64%**

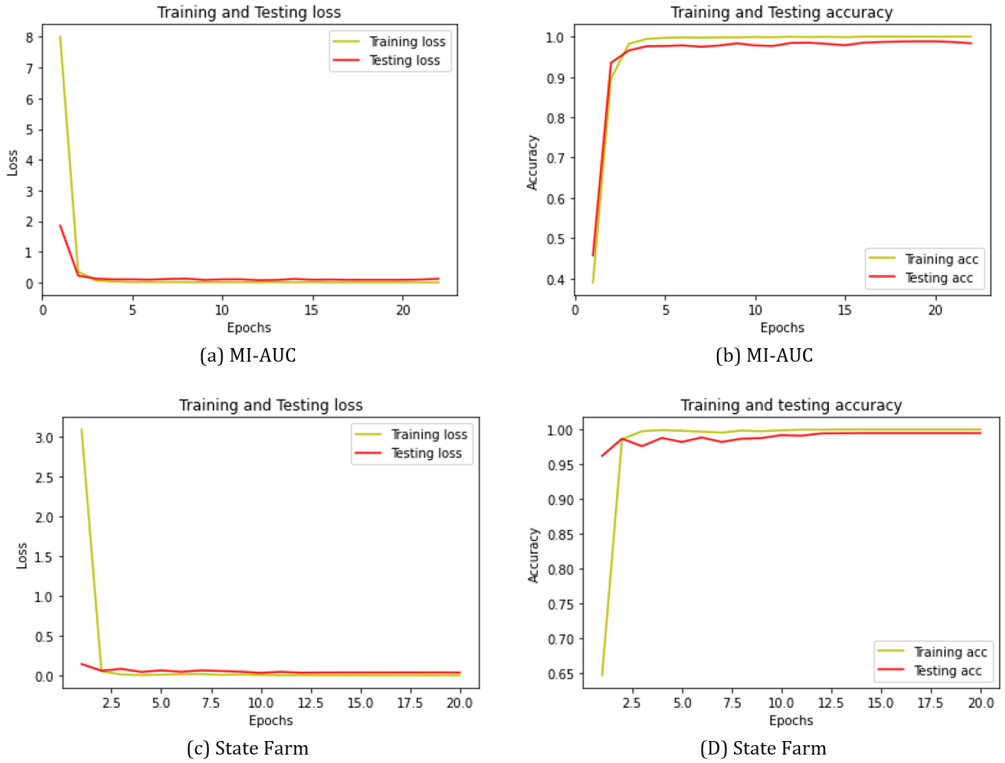

Figure 4. Accuracy and loss percentages for the U2-net model. (**a**,**b**) show the "training and testing loss", "training and testing accuracy" for the MI-AUC dataset, respectively. (**c**,**d**) show the "training and testing loss", "training and testing accuracy" for the State Farm data set, respectively.

We also compare the U2-net model's outcomes with the state-of-art CNN architectures from collected works, as shown in Table 4. The U2-net model achieves the best performance—98.34% (2.75%, 2.36%, and 2.03% higher accuracy than D-HCNN [6], GA-Weighted Ensemble, and VGG- with Regularization, respectively). In addition, Table 4 shows the approximate number of parameters for each model; the D-HCNN [6] has the lowest number of parameters (0.76 million). However, the U2-net achieves 2.75% higher accuracy than the D-HCNN model. This research focuses on enhancing the performance in terms of accuracy while enhancing the performance in terms of time complexity (considered as future work).

Suppose we observe the accuracy results from Tables 4 and 5. In that case, we can notice a gap between results generated from applying the U2-net on MI-AUC and results generated from applying the U2-net on the State Farm dataset. This gap is due to the incorrect training labels and testing data provided in the MI-AUC dataset, while the data in the State Farm dataset is accurately labeled [41].

The confusion matrix is a particular design that makes it possible to see how well a classification performs. Examples in a real class are represented in each row of the matrix, whereas those in a predicted class are represented in each column. Since the matrix's diagonal contains all the accurate predictions, it is simple to examine the accuracy directly: the numbers outside the diagonal correspond to the incorrect predictions [42]. For 10 classes of the MI-AUC and State Farm distracted driver detection datasets, an exact and comprehensive measure for the analysis of findings is presented in Figure 5 in the shape of a confusion matrix. It displays the U2-net model's categorization results. Since a large portion of the information is in the matrix's diagonal section, and there are not many misclassification mistakes, Figure 5 demonstrates the U2-net model's strong classification performance. For each class, we compute the Intersection over Union (IoU) using the confusion matrix in the manner described below. Given a confusion matrix c for N number of classes, the IoU_i for a class i is given as follows:

$$IoU_i = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{} \tag{4}$$

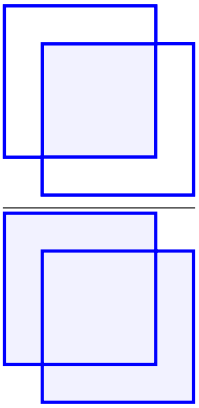

Table 5. Training and testing accuracy for the U2-Net compared to other architectures using the MI-AUC dataset [6].

Model	Image Source	Approximate No. of Parameters (in Millions) [41]	Accuracy
AlexNet [13]	Original	62	93.65%
	Skin Segmented	62	93.60%
	Face	62	86.68%
	Hands	62	89.52%
	Face + Hands	62	86.68%
Inception V3 [11]	Original	24	95.17%
	Skin Segmented	24	94.57%
	Face	24	88.82%
	Hands	24	91.62%
	Face + Hands	24	90.88%
Majority Voting Ensemble [11]	–	120	95.77%
GA-Weighted Ensemble [11]	–	120	95.98%
Original VGG [41]	Original	140	94.44%
VGG with Regularization [41]	Original	140	96.31%
Modified VGG [41]	Original	15	95.54%
Original VGG16 [43]	Original	–	79.86%
VGG16 one attention [43]	Original	–	84.82%
VGG16 two-way attention [43]	Original	–	87.74%
DenseNet + Latent + Pose [44]	Original	8.06	94.2%
MobileNet [41]	Original	4.2	94.67%
MobileNetV2 [41]	Original	3.5	94.74%
NasNet Mobile [41]	Original	5.3	94.69%
SqueezeNet [41]	Original	1.25	93.21%
Mobile VGG [41]	Original	2.2	95.24%
D-HCNN [6]	Original	0.76	95.59%
U2-net (proposed model)	Original	41.6	**98.34%**

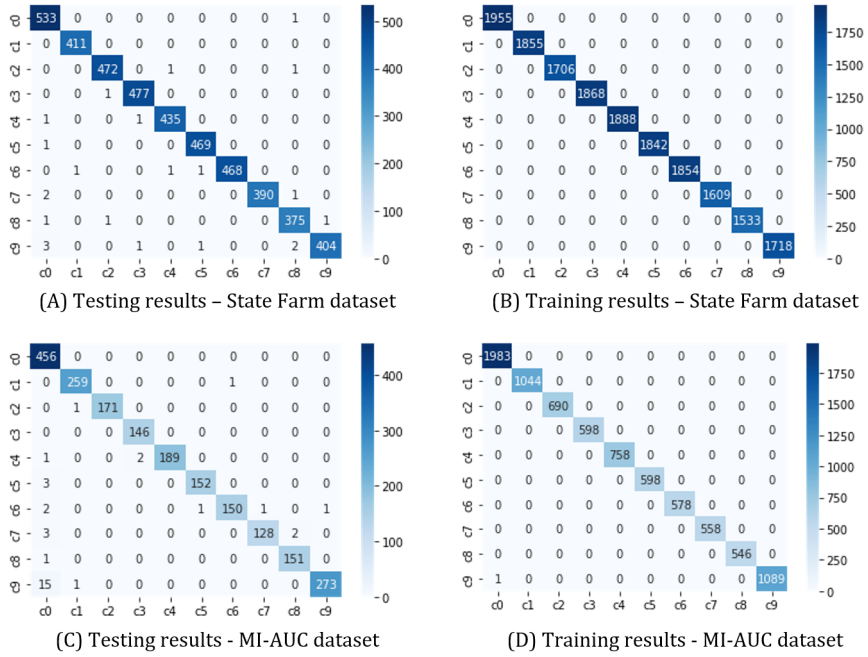

Figure 5. The confusion matrix for the U2-net model. (**A**,**B**) show the confusion matrix for the testing and training data generated from the state farm dataset, respectively. (**C**,**D**) show the confusion matrix for the testing and training data generated from the MI-AUC dataset, respectively.

Further, we use the confusion matrix to calculate the *Precision*, *Recall*, *Accuracy*, and *F-score* for each class *i* as follows. Given a confusion matrix *c* for N number of classes:

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

$$F\text{-}score = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \tag{7}$$

Tables 6 and 7 show the class-wise evaluation results for the U2-net model in terms of intersection over union, precision, recall (sensitivity), f-score, and accuracy scores for the testing data using the MI-AUC and State Farm datasets, respectively.

Table 6. Class-wise evaluation results in terms if intersection over union (IOU), Precision, Recall, Accuracy, and F-score scores using the testing part from the MI-AUC dataset.

Class	IoU	Precision	Recall Sensitivity	F-Score	Accuracy
Co—Safe Driving	94.80%	94.80%	100%	97.33%	100%
C1—Phone Right	98.85%	99.23%	99.62%	99.42%	99.62%
C2—Phone Left	99.42%	100%	99.42%	99.71%	100%
C3—Text Right	98.65%	98.65%	100%	99.32%	100%
C4—Text Left	98.44%	100%	98.44%	99.21%	98.44%
C5—Adjusting Radio	97.44%	99.35%	98.06%	98.70%	98.06%
C6—Drinking	96.15%	99.34%	96.77%	98.04%	96.77%
C7—Hair or Makeup	95.52%	99.22%	96.24%	97.71%	96.24%
C8—Reaching behind	98.05%	98.69%	99.34%	99.01%	99.34%
C9—Talking to Passengers	94.14%	99.64%	94.46%	96.98%	94.46%
Average	**97.15%**	**98.89%**	**98.24%**	**98.54%**	**98.29%**

Table 7. Class-wise evaluation results in terms if intersection over union (IOU), Precision, Recall, Accuracy, and F-score scores using the testing part from the State-Farm dataset.

Class	IoU	Precision	Recall/Sensitivity	F-Score	Accuracy
Co—Safe Driving	98.34%	98.52%	99.81%	99.16%	99.81%
C1—Phone Right	99.76%	99.76%	100%	99.88%	100%
C2—Phone Left	99.16%	99.58%	99.58%	99.58%	99.58%
C3—Text Right	99.38%	99.58%	99.79%	99.68%	99.79%
C4—Text Left	99.09%	99.54%	99.54%	99.54%	99.54%
C5—Adjusting Radio	99.36%	99.56%	99.79%	99.67%	99.79%
C6—Drinking	99.36%	100%	99.36%	99.68%	99.36%
C7—Hair or Makeup	99.24%	100%	99.24%	99.62%	99.24%
C8—Reaching behind	97.91%	98.68%	99.21%	98.94%	99.21%
C9 Talking to Passengers	98.06%	99.75%	98.30%	99.02%	98.30%
Average	**98.97%**	**99.50%**	**99.46%**	**99.48%**	**99.46%**

It is clear that the evaluation results of the U2-net model using the State Farm dataset achieve better performance results. As we mentioned earlier, this discrepancy results from some erroneous training and testing data labels given in the MI-AUC dataset, whereas the information contained in the State Farm dataset is correctly labeled [41].

Table 8 displays the assessment findings for the most cutting-edge CNN designs using the State Farm dataset in terms of the estimated total number of parameters (in millions) and accuracy. It is important to note that the U2-net paradigm outperforms these designs. It can equal and come close to the precision of D-HCNN, VGG16, and MobileVGG.

Table 8. Comparison results with the state-of-the-art approaches from literature on State-Farm dataset [6,41].

Model	Approximate No. of Parameters (in Millions)	Accuracy %
VGG16 with pretrained weights [30]	140	99.57%
VGG16 without pretrained weights [30]	140	99.43%
VGG19 with pretrained weights [30]	142	98.98%
VGG19 without pretrained weights [30]	142	99.39%
MobileVGG [41]	2.2	**99.75%**
D-HCNN [6]	0.76	**99.86%**
VGG GAP [45]	140	98.7%
CAT-CapsNet [31]	8.5	**99.88%**
DenseNet + GA [24]	–	**99.80%**
U2-net (proposed model)	41.6	**99.64%**

5. Conclusions

One of the most dangerous problems that face traffic worldwide is distracted driving. It causes a large number of fatality road accidents. We propose a new architecture, called U2-net, for detecting and classifying the state of the distracted driver while driving. The U2-net model is an extended version from the original U-net architecture [5] to automatically detect distracted drivers—especially those who are using mobile phones. The motivation behind this model is that the U2-net model uses computer vision algorithms and a deep convolutional neural network to capture the context around the objects and provide deep layers to represent image features that yield more precise classification results. It has two paths (contracting and expanding) along with a fully connected dense layer. These new paths and layers add more spatial information and context to the classifier and avoid model overfitting. To evaluate the ability of the U2 model to detect distracted drivers, we conduct experimental evaluations on two image datasets: MI-AUC and State Farm. The accuracy on MI-AUC is 98.34% and 99.64% on State Farm, which is higher than what is achieved by many other state-of-the-art models. This paper covers the distracted driver with special focus on different postures for mobile phone usage while driving. It serves

as a warning tool that alerts the driver to be aware of their behavior and make efforts to reduce distractions.

6. Future Directions

The next phases of future projects might cover detecting different postures or several causes of driver distractions, such as driver fatigue and sleepy or drowsy drivers. Additionally, another future direction might be developing more efficient models with a smaller number of training parameters to reduce the computational complexity and increase the performance accuracy. This model used images generated from inside the vehicle. Using images that are generated from outside the vehicles will be another future work or direction. Such a model might be used to alert the driver in case of distraction, which increases the drivers' awareness and their driving habits and associated risks. It also promotes safe driving practices, which in turn minimizes driver distraction and the number of crashes, therefore saving lives.

To evaluate the U2-net model, we use images generated from cameras that are inside the vehicle; however, it might be extended to accept data generated from cameras that are outside the vehicle.

Existing research gaps and challenges are:

1. Current research does not comprehensively analyze the time complexity (required time duration) to develop the system either online or offline;
2. Spatial/temporal relations should be considered. Spatial features of the current frame along with the temporal features between frames represent a crucial clue to decide the drivers' detail action. Combining the temporal and spatial information will generate higher accuracy results;
3. There is no comprehensive dataset that contains images and videos captured from different angles of driver;
4. Distracted classes are limited to some existing behaviors, New or unseen distracted behaviors that are not well-represented in the training data might result in a false positive. Adding more distracted classes and behaviors to the available datasets represents another future research direction.

Author Contributions: Conceptualization: N.O.A.; Methodology: N.O.A.; Software: N.O.A.; Validation: N.O.A. and I.A.; Formal analysis: N.O.A. and I.A.; Investigation: N.O.A.; Resources: N.O.A. and M.G.; Data curation: N.O.A. and S.A.A.; Writing (original draft preparation): N.O.A.; writing (review and editing): N.O.A. and S.A.A.; Visualization: N.O.A., I.A. and S.A.A.; Supervision: M.G.; Project administration: N.O.A.; Funding acquisition: N.O.A. All authors have read and agreed to the published version of the manuscript.

Funding: This project was partially funded from Yarmouk University Grant Number: 198/2021.

Institutional Review Board Statement: This study did not require ethical approval.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data will be available upon request.

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

References

1. Field, K.a.T. Help Nervus Drivers Transform How They Feel about Driving. 2022. Available online: https://www.confidentdrivers.co.uk (accessed on 23 November 2022).
2. ibisworld. Expert Industry Research You Can Trust. 2022. Available online: www.ibisworld.com (accessed on 23 November 2022).
3. National Highway Traffic Safety Administration. Distracted Driving. 2022. Available online: https://www.nhtsa.gov/risky-driving/distracted-driving (accessed on 23 November 2022).
4. Wu, X.; Hong, D.; Chanussot, J. UIU-Net: U-Net in U-Net for Infrared Small Object Detection. *IEEE Trans. Image Process.* **2023**, *32*, 364–376. [CrossRef] [PubMed]

5. Ronneberger, O.; Fischer, P.; Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: Proceedings of the 18th International Conference, Munich, Germany, 5–9 October 2015*; Proceedings, Part III 18; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241.
6. Qin, B.; Qian, J.; Xin, Y.; Liu, B.; Dong, Y. Distracted driver detection based on a CNN with decreasing filter size. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 6922–6933. [CrossRef]
7. Hossain, M.U.; Rahman, M.A.; Islam, M.M.; Akhter, A.; Uddin, M.A.; Paul, B.K. Automatic driver distraction detection using deep convolutional neural networks. *Intell. Syst. Appl.* **2022**, *14*, 200075. [CrossRef]
8. o. Cities, B.D. Mobile App "Maos no Volante" Hands on the Steering Wheel. 2022. Available online: https://www.paradapelavida.com.br/maos-no-volante/ (accessed on 23 November 2022).
9. Yang, J.; Sidhom, S.; Chandrasekaran, G.; Vu, T.; Liu, H.; Cecan, N.; Chen, Y.; Gruteser, M.; Martin, R.P. Detecting driver phone use leveraging car speakers. In Proceedings of the 17th Annual International Conference on Mobile Computing and Networking, Las Vegas, NV, USA, 19–23 September 2011; pp. 97–108.
10. Deshmukh, S.V.; Dehzangi, O. ECG-based driver distraction identification using wavelet packet transform and discriminative kernel-based features. In Proceedings of the 2017 IEEE International Conference on Smart Computing (SMARTCOMP), Hong Kong, China, 29–31 May 2017; pp. 1–7.
11. Eraqi, H.M.; Abouelnaga, Y.; Saad, M.H.; Moustafa, M.N. Driver distraction identification with an ensemble of convolutional neural networks. *J. Adv. Transp.* **2019**, *2019*, 4125865. [CrossRef]
12. Celaya-Padilla, J.M.; Galván-Tejada, C.E.; Lozano-Aguilar, J.S.A.; Zanella-Calzada, L.A.; Luna-García, H.; Galván-Tejada, J.I.; Gamboa-Rosales, N.K.; Velez Rodriguez, A.; Gamboa-Rosales, H. "Texting & Driving" detection using deep convolutional neural networks. *Appl. Sci.* **2019**, *9*, 2962.
13. Majdi, M.S.; Ram, S.; Gill, J.T.; Rodríguez, J.J. Drive-net: Convolutional network for driver distraction detection. In Proceedings of the 2018 IEEE Southwest Symposium on Image Analysis and Interpretation (SSIAI), Las Vegas, NV, USA, 8–10 April 2018; pp. 1–4.
14. Sultan, I. Academic Purposes 2016. Available online: https://www.kaggle.com/c/state-farm-distracted-driver-detection/discussion/20043 (accessed on 23 November 2022).
15. Sajid, F.; Javed, A.R.; Basharat, A.; Kryvinska, N.; Afzal, A.; Rizwan, M. An efficient deep learning framework for distracted driver detection. *IEEE Access* **2021**, *9*, 169270–169280. [CrossRef]
16. Kashevnik, A.; Shchedrin, R.; Kaiser, C.; Stocker, A. Driver distraction detection methods: A literature review and framework. *IEEE Access* **2021**, *9*, 60063–60076. [CrossRef]
17. Ping, P.; Huang, C.; Ding, W.; Liu, Y.; Chiyomi, M.; Kazuya, T. Distracted driving detection based on the fusion of deep learning and causal reasoning. *Inf. Fusion* **2023**, *89*, 121–142. [CrossRef]
18. Alkinani, M.H.; Khan, W.Z.; Arshad, Q.; Raza, M. HSDDD: A hybrid scheme for the detection of distracted driving through fusion of deep learning and handcrafted features. *Sensors* **2022**, *22*, 1864. [CrossRef]
19. Wang, D.; Pei, M.; Zhu, L. Detecting driver use of mobile phone based on in-car camera. In Proceedings of the 2014 Tenth International Conference on Computational Intelligence and Security, Kunming, China, 15–16 November 2014; pp. 148–151.
20. Artan, Y.; Bulan, O.; Loce, R.P.; Paul, P. Driver cell phone usage detection from HOV/HOT NIR images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, Columbus, OH, USA, 23–28 June 2014; pp. 225–230.
21. Yasar, H. Detection of Driver's mobile phone usage. In Proceedings of the 2017 IEEE 9th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management (HNICEM), Manila, Philippines, 1–3 December 2017; pp. 1–4.
22. Berri, R.A.; Silva, A.G.; Parpinelli, R.S.; Girardi, E.; Arthur, R. A pattern recognition system for detecting use of mobile phones while driving. In Proceedings of the 2014 International Conference on Computer Vision Theory and Applications (VISAPP), Lisbon, Portugal, 5–8 January 2014; Volume 2, pp. 411–418.
23. Berri, R.; Osório, F. A 3D vision system for detecting use of mobile phones while driving. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
24. Aljohani, A.A. Real-time driver distraction recognition: A hybrid genetic deep network based approach. *Alex. Eng. J.* **2023**, *66*, 377–389. [CrossRef]
25. Han, M.; Bao, Y.; Sun, Z.; Wen, S.; Xia, L.; Zhao, J.; Du, J.; Yan, Z. Automatic segmentation of human placenta images with U-Net. *IEEE Access* **2019**, *7*, 180083–180092. [CrossRef]
26. Chen, Z.; Liu, X.; Yang, J.; Little, E.; Zhou, Y. Deep learning-based method for SEM image segmentation in mineral characterization, an example from Duvernay Shale samples in Western Canada Sedimentary Basin. *Comput. Geosci.* **2020**, *138*, 104450. [CrossRef]
27. Ojeda-Pat, A.; Martin-Gonzalez, A.; Soberanis-Mukul, R. Convolutional neural network U-Net for Trypanosoma cruzi segmentation. In *Intelligent Computing Systems: Proceedings of the Third International Symposium, ISICS 2020, Sharjah, United Arab Emirates, 18–19 March 2020*; Proceedings 3; Springer: Berlin/Heidelberg, Germany, 2020; pp. 118–131.
28. Khanh, T.L.B.; Dao, D.P.; Ho, N.H.; Yang, H.J.; Baek, E.T.; Lee, G.; Kim, S.H.; Yoo, S.B. Enhancing U-Net with spatial-channel attention gate for abnormal tissue segmentation in medical imaging. *Appl. Sci.* **2020**, *10*, 5729. [CrossRef]
29. Milletari, F.; Navab, N.; Ahmadi, S.A. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In Proceedings of the 2016 fourth international conference on 3D vision (3DV), Stanford, CA, USA, 25–28 October 2016; pp. 565–571.

30. Masood, S.; Rai, A.; Aggarwal, A.; Doja, M.N.; Ahmad, M. Detecting distraction of drivers using convolutional neural network. *Pattern Recognit. Lett.* **2020**, *139*, 79–85. [CrossRef]
31. Mittal, H.; Verma, B. CAT-CapsNet: A Convolutional and Attention Based Capsule Network to Detect the Driver's Distraction. *IEEE Trans. Intell. Transp. Syst.* **2023**, *24*, 9561–9570. [CrossRef]
32. Abbas, T.; Ali, S.F.; Mohammed, M.A.; Khan, A.Z.; Awan, M.J.; Majumdar, A.; Thinnukool, O. Deep Learning Approach Based on Residual Neural Network and SVM Classifier for Driver’s Distraction Detection. *Appl. Sci.* **2022**, *12*, 6626. [CrossRef]
33. Khan, T.; Choi, G.; Lee, S. EFFNet-CA: An Efficient Driver Distraction Detection Based on Multiscale Features Extractions and Channel Attention Mechanism. *Sensors* **2023**, *23*, 3835. [CrossRef]
34. Yin, X.X.; Sun, L.; Fu, Y.; Lu, R.; Zhang, Y. U-Net-Based Medical Image Segmentation. *J. Healthc. Eng.* **2022**, *2022*, 4189781. [CrossRef]
35. Wambura, S.; Li, H.; Nigussie, A. Fast memory-efficient extreme events prediction in complex time series. In Proceedings of the 2020 3rd International Conference on Robot Systems and Applications, Chengdu, China, 14–16 June 2020; pp. 60–69.
36. Roth, H.R.; Lu, L.; Liu, J.; Yao, J.; Seff, A.; Cherry, K.; Kim, L.; Summers, R.M. Improving computer-aided detection using convolutional neural networks and random view aggregation. *IEEE Trans. Med. Imaging* **2015**, *35*, 1170–1181. [CrossRef]
37. Chollet, F. *Deep Learning with Python*; Simon and Schuster: New York, NY, USA, 2021.
38. Wambura, S.; Li, H. Deep and confident image analysis for disease detection. In Proceedings of the 2020 2nd International Conference on Video, Signal and Image Processing, Jakarta, Indonesia, 4–6 December 2020; pp. 91–99.
39. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. *arXiv* **2014**, arXiv:1412.6980.
40. Funt, B.; Zhu, L. Does colour really matter? Evaluation via object classification. In Proceedings of the Color and Imaging Conference. Society for Imaging Science and Technology, Vancouver, BC, Canada, 12–16 November 2018; Volume 26, pp. 268–271.
41. Baheti, B.; Talbar, S.; Gajre, S. Towards computationally efficient and realtime distracted driver detection with mobilevgg network. *IEEE Trans. Intell. Veh.* **2020**, *5*, 565–574. [CrossRef]
42. Wikipedia. Confusion Matrix. 2022. Available online: https://en.wikipedia.org/wiki/Confusion_matrix (accessed on 23 November 2022).
43. Ai, Y.; Xia, J.; She, K.; Long, Q. Double attention convolutional neural network for driver action recognition. In Proceedings of the 2019 3rd International Conference on Electronic Information Technology and Computer Engineering (EITCE), Sanya, China, 30–31 October 2019; pp. 1515–1519.
44. Behera, A.; Keidel, A.H. Latent body-pose guided densenet for recognizing driver's fine-grained secondary activities. In Proceedings of the 2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), Madrid, Spain, 29 November–2 December 2018; pp. 1–6.
45. Zhang, B. *Apply and Compare Different Classical Image Classification Method: Detect Distracted Driver*; Project Report; Department Computer Science, Stanford University: Stanford, CA, USA, 2016; pp. 1–6.