International Journal of Network Dynamics and Intelligence Scilight

Article

Background Replacement in Video Conferencing

Kiran Shahi*, and Yongmin Li

Department of Computer Science, Brunel University London, Uxbridge, UB8 3PH, United Kingdom
* Correspondence: 2144420@alumni.brunel.ac.uk

Received: 17 March 2023
Accepted: 24 April 2023
Published: 23 June 2023

Abstract: Background replacement is one of the most used features in video conferencing applications by many people, perhaps mainly for privacy protection, but also for other purposes such as branding, marketing and promoting professionalism. However, the existing applications in video conference tools have serious limitations. Most applications tend to generate strong artefacts (while there is a slight change in the perspective of the background), or require green screens to avoid such artefacts, which results in an unnatural background or even exposes the original background to other users in the video conference. In this work, we aim to study the relationship between the foreground and background in real-time videos. Three different methods are presented and evaluated, including the baseline U-Net, the lightweight U-Net MobileNet, and the U-Net MobileNet&ConvLSTM models. The above models are trained on public datasets for image segmentation. Experimental results show that both the lightweight U-Net MobileNet and the U-Net MobileNet& ConvLSTM models achieve superior performance as compared to the baseline U-Net model.

Keywords: video conferencing; background replacement; image segmentation; U-Net; mobilenet; ConvLSTM

1. Introduction

With the increase in work from home (WFH) during the COVID pandemic, the video conference tools like Zoom, MS Teams and Skype have been used more than ever. These platforms have played significant roles in replacing the traditional physical-office-based work with the virtual-office-based work. The increasing use of these platforms highlights severe privacy issues like the background exposure in sensitive environments. To avoid users' privacy leakage, the background replacement concept is introduced. This feature enables users to hide their sensitive or private background information by blurring or replacing it with a virtual background. The background replacement technique involves separating the image into two or more layers (generally into the foreground and background) and merging the separated foreground with the new virtual background. In this case, the target in the foreground is normally one person or a group of people, while the background is the surroundings to be replaced.

The background replacement techniques have been widely used in video conferencing tools like Microsoft Teams, Skype, Zoom and Google Meet. However, there are still serious limitations even with these popular video conferencing platforms. First, the current virtual background implementations do not work accurately along the boundaries between the foreground and background. This produces artefacts at the boundaries, renders an unnatural virtual background, and could even expose the real background. Second, most of the current applications perform the background replacement using static methods, i.e. dealing with individual images instead of continuous videos. At times, a slight movement of the subject may cause significant changes and make the virtual background inconsistent. Therefore, it remains a challenging task to estimate the foreground accurately and consistently in these settings.

These challenges motivate us to study the relationship between the background and foreground layers concerning the artefact generated during segmentation. In this paper, we will present three different methods for background replacement of real-time videos from a webcam, including 1) a baseline U-Net; 2) a lightweight U-Net MobileNet; and 3) a U-Net MobileNet & ConvLSTM. The main contributions of this paper are summarised as follows:

(1) A lightweight encoder-decoder-based architecture (that uses MobileNetV3-Large as the feature extractor in the encoder block and the residual convolution in the decoder block) is developed in this work. This network can operate 14 frames per second while being tested via NVIDIA GeForce GTX 1050 Ti with 4GB GPU Memory, which is almost double faster than the baseline U-NET method.

(2) A U-Net MobileNet&ConvLSTM method is presented which can utilize the temporal information in the continuous video input. The proposed method performs much better than the other two methods while being tested on real-time videos through a webcam. The result is presented in Figure 1.

These methods are robust enough to replace the background while being tested on real-time videos via a webcam. However, we found that it is very challenging to predict the target object for segmentation in the cluttered background with low illumination, see Figure 1.

The rest of the paper is organised as follows. First, we review in Section 2 the previous studies on image matting and segmentation, and the key issues of replacing backgrounds in video conferencing. The proposed methods are presented in Section 3. Experiments and result analysis are provided in Section 4 before conclusions are drawn in Section 5.

2. Background

The core of background replacement is how to separate the foreground subjects from the background surrounding. The existing methods for background subtraction are pretty good at prediction when the auxiliary input is supplied as a parameter to guide the target object. However, due to the dynamic nature of the surroundings in real-time applications like video conferencing, it is not feasible to supply an extra image to guide the segmentation. Similarly, the slight movement in the camera angle or target object reveals the background, which leads to an inconsistent result. In this section, we will review the previous work on this topic from the traditional matting and segmentation methods, to the most recent deep learning based approaches. Meanwhile, we will also review the dynamic models that address the temporal characteristics of continuous video inputs.

2.1. Matting

Matting is the process of separating the image into two or more layers (generally into the foreground, background and mask), or separating the alpha image that determines the blending layers of two or more image elements into a single image or a frame [1].

Mathematically, it can be defined as $I = \alpha F + (1-\alpha)B$, where I is a given frame (image), F is the foreground, B is the background, and α is the alpha matte. Therefore, the frame I is a linear combination of the foreground F and background B through a coefficient α [2]. Trimap-based matting is one of the most used methods in computer vision [3]. In this method, the trimap is provided as an auxiliary input with three regions: the foreground, background and unknown [4].

Xu et al. [5] developed a deep convolutional network that consists of two distinct models. The first model is an encoder-decoder architecture that predicts an alpha matte of an image, and the second model is a convolutional network that refines the alpha matte prediction of the first network with more accurate alpha values and sharper edges. This method achieves the SAD score of 50.4 and MSE of 0.014 with a dataset created and published containing 49300 training and 1000 testing images. Most of the recent matting research uses this dataset to train their model. A similar encoder-decoder architecture was presented in [6], which uses the ResNet-50 trained on the ImageNet dataset as an encoder. The SAD and MSE score of this method are 25.8 and 0.0052, respectively. Using ResNet-50 as an encoder has improved the performance significantly. Although the trimap base method performs well, it requires the manual trimap as an extra input. As it is impossible to provide the trimap manually for real-time videos, this method seems unfeasible to real-time videos. So, this method is out of our scope.

Sengupta et al. [7] proposed an adversarial network, where generator G is a deep matting network that extracts the foreground colour and alpha from the input image, and discriminator D guides the training to generate realistic images. Here, the generator network (G) is a residual encoder-decoder neural network. This method produces a SAD score of 1.72 and an MSE of 0.97e-2 on the Adobe image matting (AIM) dataset. Further, Lin et al. [8] proposed BackgroundMattingV2, which is an enhanced version of the background matting method [7]. In this method, ResNet-50 is adopted as the encoder block of the generator and the ASPP (the atrous spatial pyramid pooling) block after the backbone. This method achieves a SAD score of 1.286 and an MSE of 12.01e-3, and obtains better results on AIM datasets than the previous methods. Hence, the prediction accuracy is improved when using ResNet-50 (pre-trained on the ImageNet dataset) as a backbone. While analysing the metrics of the background matting method against the trimap based method, it seems that the background matting method has better accuracy compared to the trimap based method. However, this method requires an additional input image without the subject (human). Therefore, its application to video conferencing is limited.

2.2. Segmentation

Image segmentation is the process of grouping similar regions or segments of an image under their respective object type. In image matting, the value of the alpha matte is constrained within (0, 1); however, in image segmenta-

tion, it is either 0 or 1 [9]. Therefore, image segmentation generates a binary image where each pixel belongs to either the foreground or background.

Ronneberger et al. [10] proposed a convolutional network architecture (called the U-Net), particularly for biomedical image segmentation. It consists of a contracting path to capture the context and a symmetric expanding path that enables precise localization. The feature map in the contracting path is cropped and concatenated with the corresponding up-sampled feature maps. The cropping helps to propagate contextual information along the network and allows to segment objects in an area using the context from a larger overlapping area. Considering humans as the subject, Xie et al. [11] published a study on the effectiveness of the U-Net on foreground segmentation. In this research, the authors achieved a 91% accuracy rate, which proves that the U-Net architecture could be used for foreground segmentation. However, the research has left lots of spaces for future work. Taking advantage of these capabilities of the U-Net, Kuang and Tie [12] proposed a flow-based video segmentation method for human heads and shoulders, which is called FUNet. With this method, the authors achieved an average dice coefficient of 0.96 that seems good, but there are still spaces for improvement to achieve better accuracy while evaluating the qualitative results of the model.

The U-Net is widely used in segmentation tasks, and various extensions of the U-Net are developed to solve domain-specific problems. Zhang et al. [13] introduced the residual U-Net, where the problem of vanishing or exploding gradients (due to complex and deep neural networks) is solved via introducing residual blocks instead of simple convolutional blocks. The residual U-Net is initially proposed for road extraction from high-resolution aerial images in remote sensing image analysis, and is later applied to polyp segmentation, brain tumour segmentation, etc. [13].

Liang et al. [14] introduced a segmentation method called the TriSeNet. Unlike the U-Net, the whole network in this method is composed of three different network paths to extract the high-dimensional spatial features, high-level semantic features and detailed boundary features. This network is trained and tested on an MSSP20K dataset with an IoU score of 90.43%. This method is trained to predict a single person by aggregating multiple cues, but fails when there are multiple subjects. Also, the method fails to predict the target subject in challenging scenes like shadows.

The methods mentioned above typically have large numbers of training parameters and require high-end computing resources. Miao et al. [15] developed the PSPNet-50 to address this issue, where the convolutional-layer level pruning technique is used to optimise the network size and parameters. After optimisation, the parameters are reduced from 46.7 M to 6.2 M, and the FPS increases from 13 to 31. However, the model's accuracy decreases from 94.8% to 93.2%. The comparisons with the base model is also made before pruning and after pruning. While reviewing this method's qualitative and quantitative evaluation, we realise that further work could be done to optimise this network.

Zhang et al. [16] developed the PortraitNet, which is a real-time portrait segmentation network for the mobile device, where a lightweight backbone (MobileNet-V2) is used as an encoder to achieve a high inference speed. The network parameter is 2.1 M which is significantly low compared to the previous methods. The mean IoU is 93.43% on the supervise.ly dataset.

2.3. Dynamic Models

The aforementioned studies mostly address the problem statically, either taking input from still images or treating each video frame as an independent image. To address the dynamic nature of the problem, the recurrent neural network based methods could be the best alternative while dealing with sequential data from a video input. The ConvLSTM [17] and ConvGRU [18] are two recurrent architectures that are adopted from the long short-term memory (LSTM) and the gated recurrent unite (GRU).

Bearing this in mind, the concept of using frames of an input video as a data sequence was introduced in [19], where the recurrent architecture is used to exploit temporal information in videos which can significantly improve temporal coherence and matting quality. This architecture comprises an encoder that extracts individual frame features, a recurrent decoder that aggregates temporal information, and a deep guided filter module for high-resolution upsampling. To extract the temporal information, the ConvGRU is integrated on the half channel of each layer in the decoder. While working well on the simple background such as a green screen, the major drawback of this method is that it fails to predict the target object in the complex background.

3. Methods

Based on the encoder-decoder architecture of the U-Net, we will develop three different network models in this study, including the baseline U-Net, U-Net MobileNet and U-Net MobileNet&ConvLSTM. The architectures of these models are illustrated in Figures 2, 3 and 4.

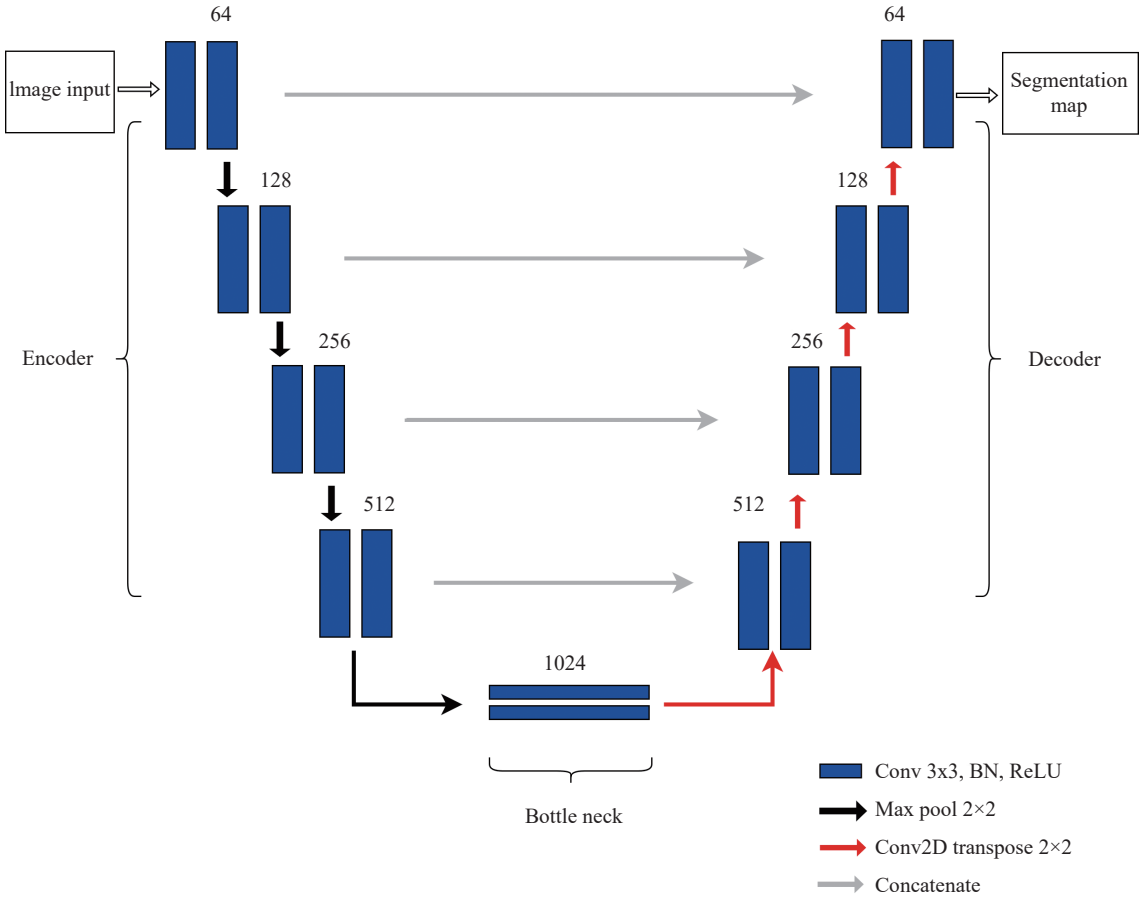

Figure 2. Network architectures of the baseline U-Net model.

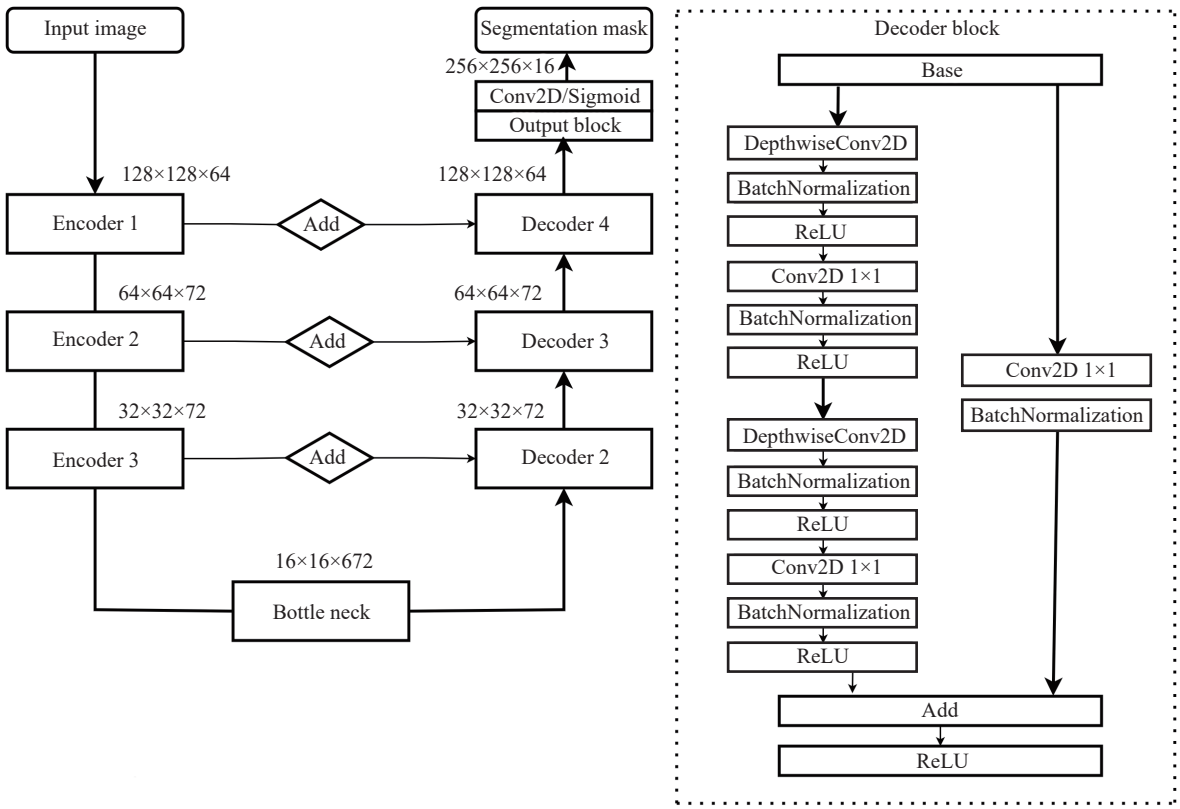

Figure 3. Network architectures of the lightweight U-Net MobileNet model.

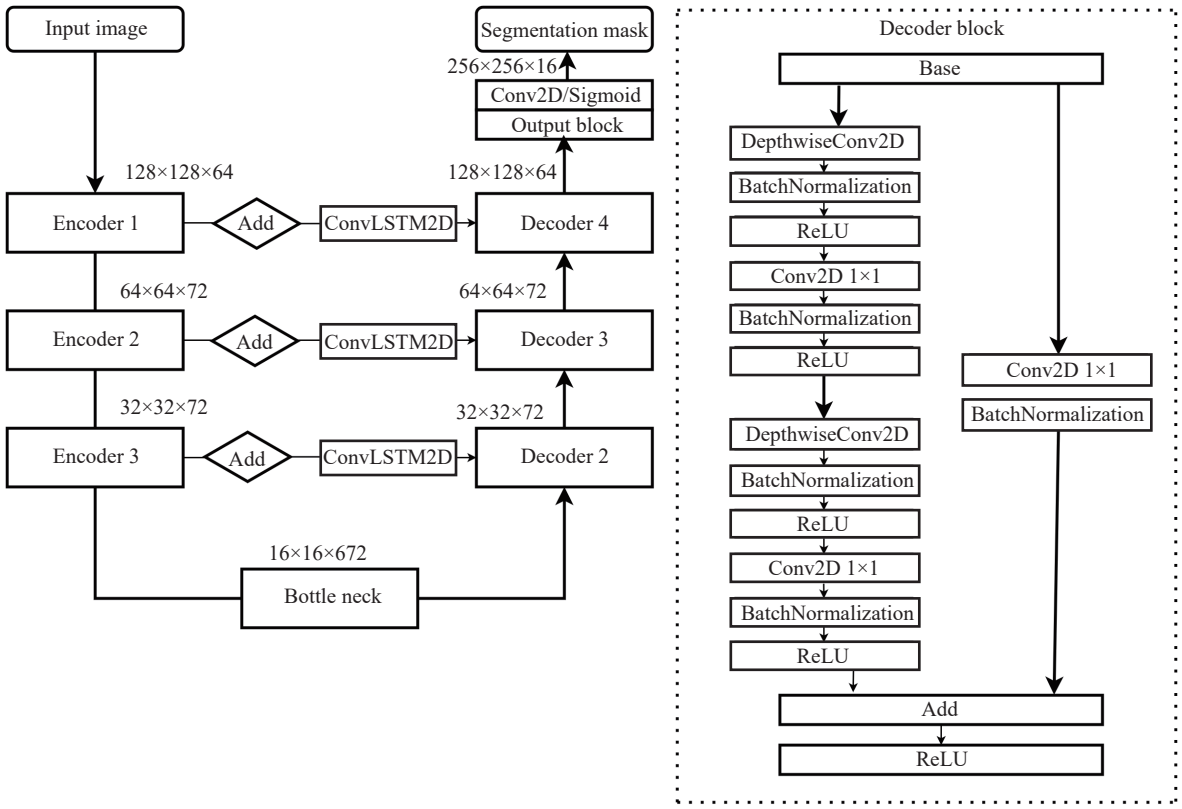

Figure 4. Network architectures of the U-Net MobileNet & ConvLSTM model.

3.1. U-Net

For the first experiment, we implement the U-Net architecture and intend it as a baseline model. In this architecture, there are two paths. The first path is the contraction path, known as the encoder, and the second path is the expanding path, known as the decoder. Here, the role of the encoder is a features extractor that learns the abstract representation of the input image through a sequence of the encoder blocks. The decoder helps to get more complex features at the loss of localization information. The localization information is obtained from the encoder path. We adopt the base U-Net that operates at the $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ and $\frac{1}{16}$ feature scales. Each block in the encoder and decoder sections is the combination of 3x3 convolution layers followed by the batch normalization and the ReLU activation function.

3.2. U-Net with MobileNetV3-Large as Backbone (U-Net MobileNet)

In this network, we have used the MobileNetV3-Large as the feature extractor due to the fact that, the MobileNetV3-Large is light weight with less parameters and still manages to achieve better performance on the ImageNet dataset. For the encoder path, we pick the 16th, 24th, 38th and 193rd layer. Basically, these are ReLU activation layers. For the decoder path, we use a residual block. In our residual block, we pass the previous layer output to two blocks. One block has a couple of convolution blocks made up of the depth-wise convolution, batch normalization, ReLU, Conv2D, and ReLU activation function. The other block is simply a convolution layer with Conv2D and batch normalization. Again, we add the two layers as the skip connection that is passed through the ReLU activation function. Furthermore, the residual block is fed to the next decoder block.

3.3. U-Net with MobileNetV3 as Backbone and ConvLSTM (U-Net MobileNet & ConvLSTM)

Inspired by the bi-directional ConvLSTM U-Net with the densely connected convolution by Azad et al. [20], the experiment is conducted by passing the tensor to the ConvLSTM layer after merging encoder and decoder blocks with the skip connection. The idea behind using the ConvLSTM between the encoder and decoder is to capture the temporal information from a continuous video input, and also to act as a mechanism to learn non-linear representation shared among encoding and decoding paths. The feature set from the encoder path has more localization information, whereas the decoder path brings the semantic information.

4. Experiments

The details of the experiments, including the datasets, loss functions, model training, results and analysis are described as follows.

4.1. Data and Augmentation

We have used the Person segmentation dataset [21] to train our model. To improve the generalization and robustness of our model, we have applied various data augmentation techniques to the above mentioned dataset. The augmentation technique helps to artificially increase the amount of data by generating new data points from the existing data, increase the model generalisation capabilities, and prevent overfitting. We perform the following operations to augment our data: the horizontal flip, colour change (RGB to gray-level), channel shuffle, coarse dropout (min holes=3, max holes=10, max height=32 and max width=32), and rotation 45°. After augmentation, we split each dataset by 80:10:10 into the train, test and validation datasets.

4.2. Loss Function

The two widely used loss functions in semantic segmentation are the cross entropy and dice loss functions. The cross-entropy loss function evaluates the class prediction for each pixel individually, and then calculates the averages of overall pixels. Nonetheless, such a function can cause a problem if we have an unbalanced representation of the image, as the most prevalent class can dominate the training performance. In contrast, the dice loss function evaluates the intersection and union over the foreground pixel, which can deal with the issue of the unbalanced class.

For example, in our case, we have an image of a human and want to segment our image as the foreground (the human) and the background (not the human). In most of these images, we likely see most of the pixels in an image that is not the human. On average, we may find that 70-90% of pixels in the image correspond to the background and only 10-30% of pixels correspond to the foreground.

If we use the cross-entropy loss function, the algorithm may predict most of the pixels as the background even when they are not, and this still produces low errors. In the case of the dice loss function, if the model predicts all the pixels as the background, the intersection would be 0, and this gives rise to an error of 1. Hence, we use the dice loss function as the loss function. We apply the dice loss function to understand the relation between the foreground A with respect to the ground-truth B. The dice coefficient that measures the overlap between two samples A and B can be calculated as:

$$Dice\ Coefficient = \frac{2|A \cap B|}{|A| + |B|} \tag{1}$$

$$Dice\ Loss = 1 - \frac{2|A \cap B|}{|A| + |B|} \tag{2}$$

Here, $A \cap B$ is the common region between the predicted foreground A and the ground truth B, and $A + B$ represents the elements in A and B. The coefficient value ranges from 0 to 1, where 1 denotes the perfect prediction and complete overlap, and 0 is the opposite [22, 23].

4.3. Hyperparameters Setup

Our models are trained via the functional API of TensorFlow using the Adam optimizer. The learning rate is initialized at 1e-4 and the decay rate is set as 0.1 for every 5 epochs until the minimum learning rate reaches 1e-7. Similarly, we set a batch size of 8 and resize each image to 256 x 256 pixels to reduce the computation overhead. The epoch is initialized at 100, the early stopping is set as 10 epochs, and the batch-normalization layers are added after every convolution layer with a default decay rate of 0.99.

4.4. Training

U-Net: The baline model is trained on the Person segmentation dataset for 68 epochs with a batch size of 8. Initially, we start the training with a learning rate of 1e-4 and monitor the validation loss. We decrease the learning rate by a factor of 0.1 until it reaches 1e-7. Up to 32 epochs, the model is trained at a learning rate of 1e-4, and the best validation loss is 0.04669. We reduce the learning rate to 1e-5, and after this modification of the learning rate, the validation loss and other evaluation metrics are slightly improved. However, the improvement in learning stops from the 52^{nd} epoch, and we reduce the learning rate at the 57^{th} epoch to 1e-6. After reducing the learning rate, the loss does not improve, so we reduce the learning rate to 1e-7 after five more epochs. Also, with a learning rate of 1e-7, the learning capabilities do not improve, so the training stops at the 69^{th} epoch.

U-Net MobileNet: This model is trained for 46 epochs where each epoch takes almost 6 minutes. During training, we monitor the validation loss and reduce the learning rate when the metric stops improving for four epochs. Similarly, we configure the early stopping at the patience of 10 with validation loss, which means that the training will be stopped if the learning does not improve for the last ten epochs. Further, we configure the model checkpoint to save only the best performing epoch.

While training the model, the validation loss fluctuates at a high rate until 19 epochs, but the training loss is constantly improving. However, the best learning rate (with a learning rate of 1e-4) is 0.04292 at epoch 15. We fur-

ther train the model by reducing the learning rate by 0.1, i.e., 1e-5. After reducing the learning rate to 1e-5, the model improves further to validation loss 0.02406. Further, the learning rate is decreased to 1e-6 at the 31^{st} epoch, and the validation loss slightly improves to 0.02298. The best validation loss (obtained at the learning rate of 1e-6) is 0.02298 at the 32^{nd} epoch. However, the validation loss does not improve further. Then, the learning rate is decreased to 1e-7, and the validation loss is improved slightly to 0.02297 at the 36^{th} epoch. Similarly, the learning rate is further decreased to 1e-9 to reduce the validation loss. Again, the validation loss does not improve further. Hence, we restore the best performance model at the 36^{th} epoch and save it as the final model.

U-Net MobileNet&ConvLSTM: This network is trained for 44 epochs on the Person segmentation dataset. Initially, we start the training with a learning rate of 1e-4, and it takes approximately 10 minutes per epoch. We monitor the validation loss and reduce the learning rate when the metric stops improving for four epochs. Similarly, we configure the early stopping at the patience of 10 with validation loss, which means that the training will be stopped if the learning does not improve for the last ten epochs. Further, we configure the model checkpoint to save only the best performing epoch.

During the training, the dice loss on the validation set fluctuates at an extremely high rate in the first few epochs. With a learning rate of 1e-4, the best validation dice loss that we could achieve is 0.08928 at the eighth epoch. The learning rate does not improve until the 12^{th} epoch, and we reduce the learning rate to 1e-5 at the 13^{th} epoch. After reducing the learning rate, the validation loss stabilises at a specific rate. The best performing epoch with a learning rate of 1e-5 is at 19^{th} with a validation loss of 0.02773. However, the overall best performing model is at the 34^{th} epoch with a dice loss of 0.02604, and is saved as the final model.

4.5. Results

For quantitative evaluations, we first evaluate the dice loss, dice coefficient, mean absolute difference and mean squared error on the Kaggle Person segmentation dataset [21]. Table 1 shows the performance on the test data of Person segmentation.

Table 1 Performance comparison among the three methods on the Kaggle Person segmentation dataset [21]

Method	Dice Loss	Dice Coef	MAD	MSE
U-Net	0.0258	0.9741	0.0193	0.0190
U-Net MobileNet	**0.0254**	**0.9745**	**0.0182**	**0.0176**
U-Net MobileNet & ConvLSTM	0.0274	0.9725	0.0209	0.0207

The results indicate that, although all three methods perform well on the same dataset, the performance of the U-Net architecture with MobileNetV3 as the backbone marginally outperforms that of the other two methods.

Table 2 lists the parameters, model size and speed frames per second (FPS). It is clearly shown that the baseline U-Net model is the heaviest model which is computationally expensive, and the other two models are considerably lightweight and fast. We test the FPS of each model on a computer with NVIDIA GeForce GTX 1050 Ti of 4GB GPU Memory.

Table 2 Model complexity (number of parameters and model size) and the real-time performance in frames per second (FPS)

Method	Parameters	Size (MB)	FPS
U-Net	31, 055, 297	355	8
U-Net MobileNet	**1, 033, 545**	**12.9**	**14**
U-Net MobileNet & ConvLSTM	2, 094, 281	25.1	9

Similarly, to understand the generalization ability of our model, we evaluate our model by using the unseen dataset (Conference Video Segmentation Dataset [12]) published by Kuang and Tie. Table 3 illustrates the performance of our model on the test set of the Conference Video Segmentation Dataset dataset.

Table 3 Performance comparison among the three methods against FUNet on the ConferenceVideoSegmentation-Dataset [12]

Method	Dice Coef
U-Net	0.9665
U-Net MobileNet	**0.9692**
U-Net MobileNet & ConvLSTM	0.9680
FUNet	0.9600

Table 3 shows that our method performs significantly well on the unseen datasets, and this indicates that the U-Net MobileNet method performs superior to other methods.

A sample of the segmentation results from the Kaggle Person segmentation dataset is shown in Figure 5, and such results are consistent with the results in Table 1 in that all three methods perform well.

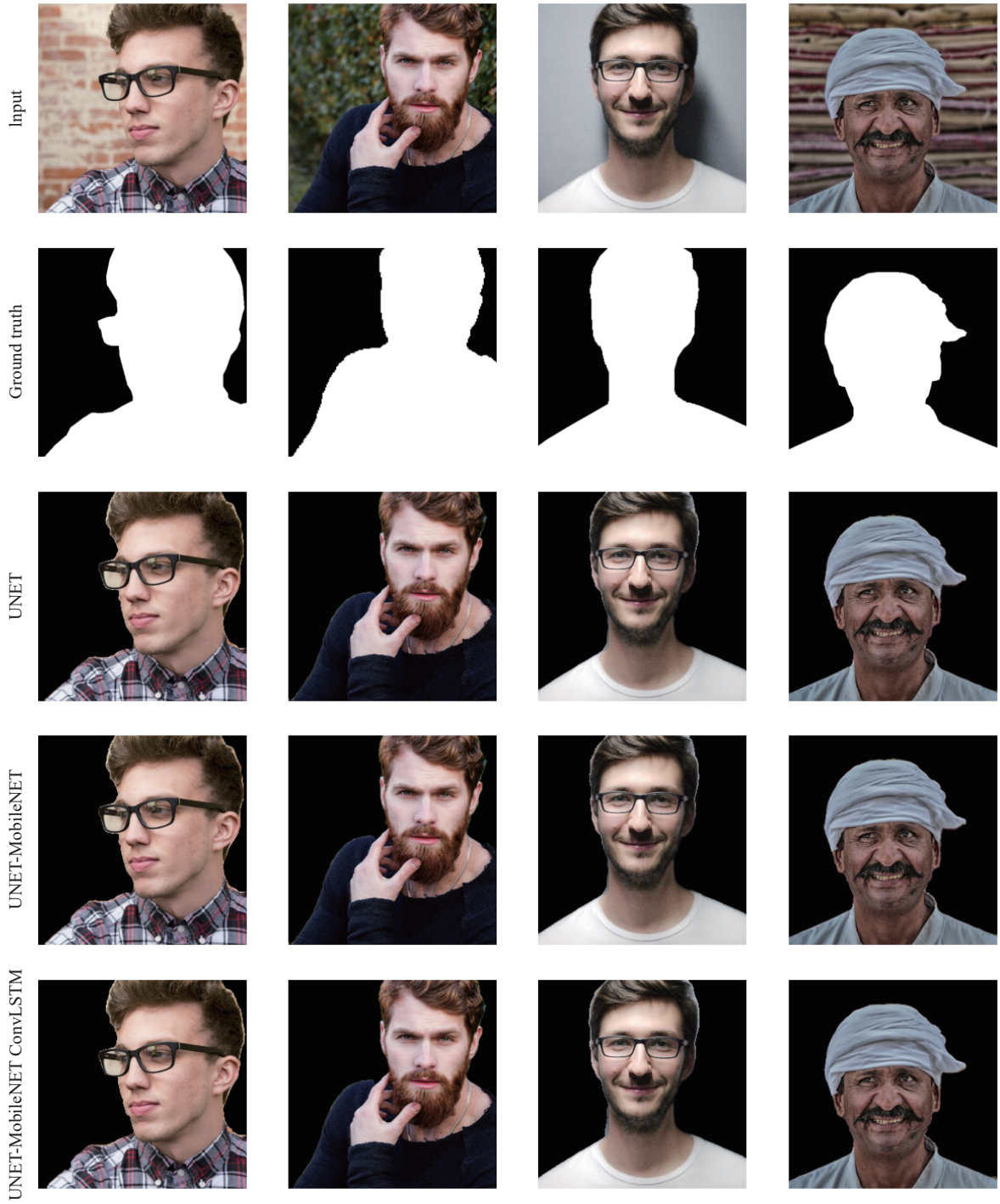

Figure 5. Results on the Kaggle Person Segmentation dataset [21].

We also test the models on our own videos recorded from a webcam which is significantly different from the images in the Kaggle dataset [21]. Figure 6 and Figure 1 show two examples where the former has a clean background and the latter has a cluttered background. Since we do not have the ground-truth segmentation results, comparisons can only be done through a visual evaluation.

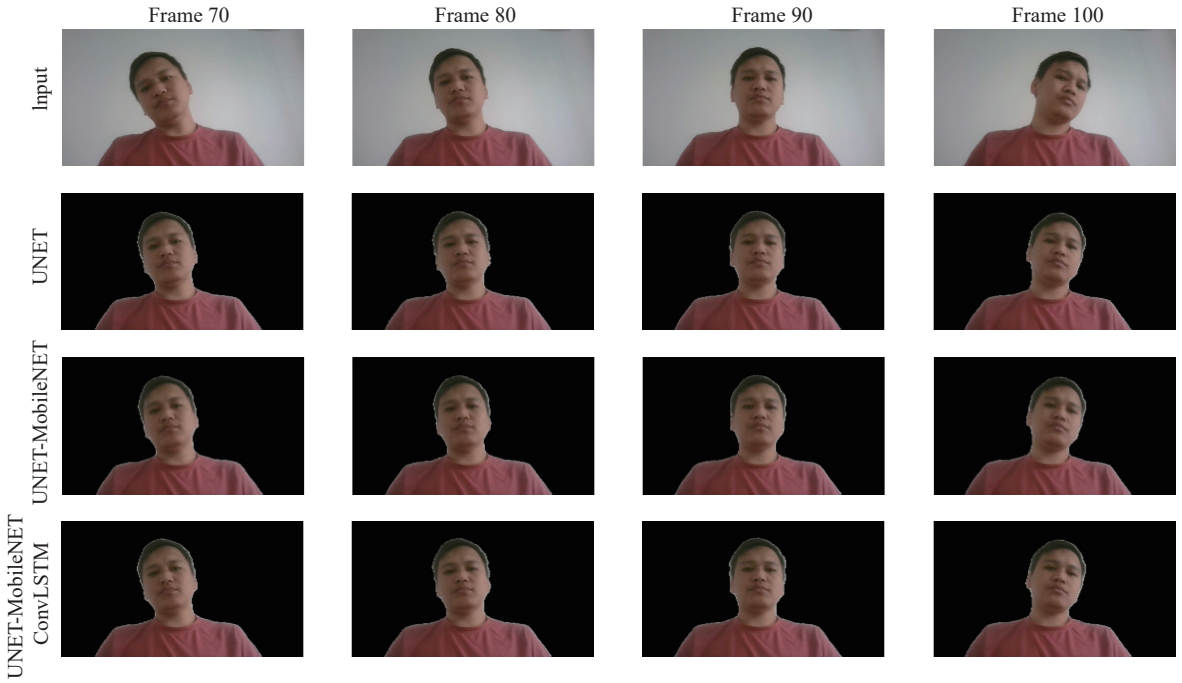

Figure 6. Results on new video with a clean background.

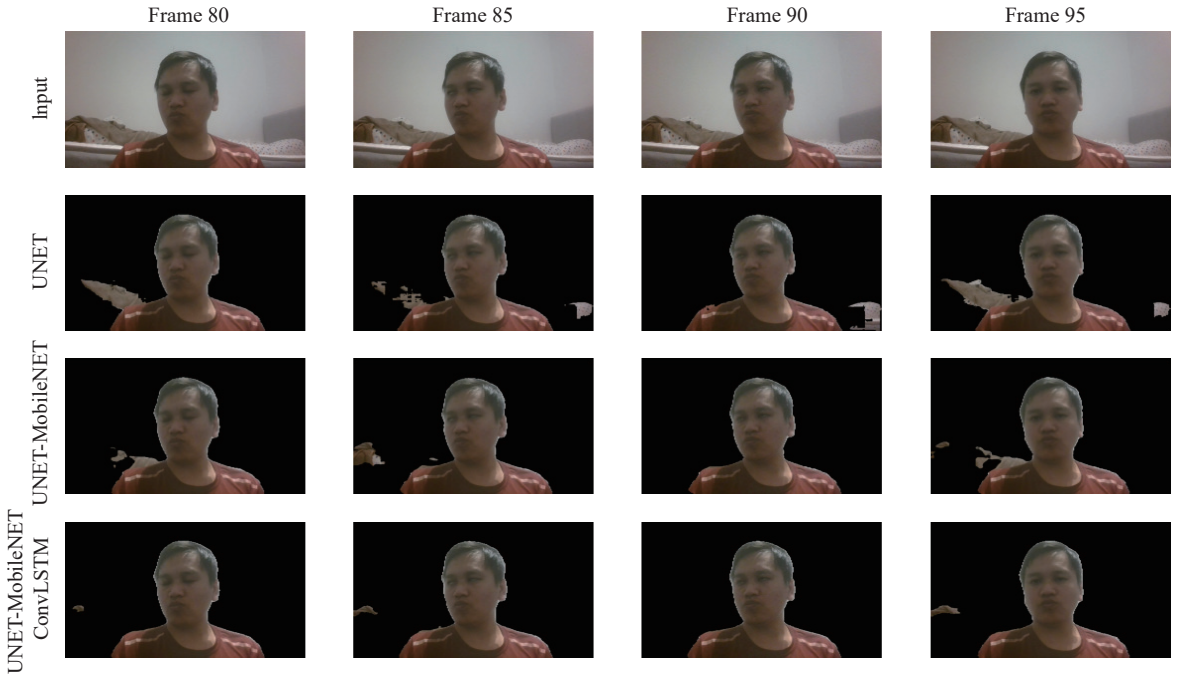

Figure 1. Results on new video with a cluttered background.

Again, all three methods perform well on the clean background as shown in Figure 6. However, for the cluttered background in Figure 1, the baseline U-Net model performs poorly, while the other two models produce much better segmentation. In particular, for the U-Net MobileNet&ConvLSTM, when the temporal information from continuous frames is processed, the segmentation results are more consistent and cleaner.

Furthermore, we plot the output of our experiments against the ConferenceVideoSegmentationDataset dataset to visualise the performance metrics presented in Table 3. The results indicate that the performance of the baseline U-Net method is significantly poorer compared to the other three methods. Though the other three methods perform significantly well, we can see a few incorrect predictions by the FUNet and U-Net MobileNet&ConvLSTM if we focus on tiny details around the arms Figure 7.

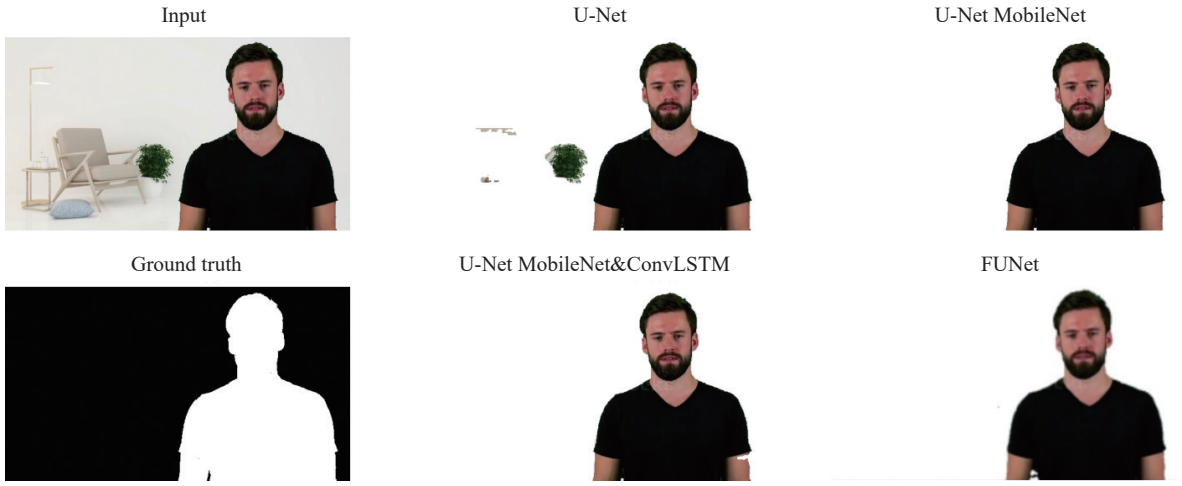

Figure 7. Results on the ConferenceVideoSegmentationDataset [12].

5. Conclusions

To address the problem of background replacement in video conferencing, we have developed three U-Net based models in this work including 1) the baseline U-Net; 2) the lightweight U-Net MobileNet; and 3) the U-Net MobileNet&ConvLSTM. The baseline U-Net, as expected, is the heaviest with the highest number of parameters and the slowest running speed. The key ideas behind the development of the lightweight U-Net MobileNet and the U-Net MobileNet&ConvLSTM models are to reduce the computation, and include the temporal information from continuous video inputs into the process of background segmentation.

We have trained the models on the public dataset of Kaggle Person segmentation [21], and tested them on both the aforementioned dataset and our own recorded videos. From the experimental results, we can reach the following conclusions.

(1) All three methods perform well on easy cases with relatively clean backgrounds. On the test split of the public dataset, the lightweight U-Net MobileNet has marginally better performance by all the metrics used in the experiments including the dice loss, dice coefficient, MAD and MSE.

(2) Both the U-Net MobileNet and the U-Net MobileNet&ConvLSTM have very low number of parameters and run faster than the baseline U-Net model.

(3) Our own test videos recorded from a webcam indicate that the lightweight U-Net MobileNet and the U-Net MobileNet&ConvLSTM models perform much better than the baseline model, where temporal information from continuous video inputs has been included in the U-Net MobileNet&ConvLSTM model.

The presented method can be adopted in both industry and academia. In academia, our segmentation technique can be adopted for research in medical imaging, like retinal blood vessel segmentation and brain tumor segmentation. In industry, a zoom plugin that replaces the background can be built on the top of our model. Similarly, the presented method can be implemented as a filter in social applications such as Snapchat and Instagram.

Though positive results have been produced, much work can be done to further improve the performance. For example, our methods do not perform well when there is constant movements in the background. Further, other architectures for semantic segmentation can be explored such as the MobileNet and ConvLSTM that are presented in this paper.

Our research shows that the utilisation of the temporal information that is present in continuous frames can improve the robustness of the model significantly. Due to the nature of our dataset, we perform one-to-one recurrent convolution. Therefore, further studies will be directed towards the temporal coherence in video frames by using the state-of-art neural network methods like the attention mechanisms.

Author Contributions: Yongmin Li: Conceptualization; **Kiran Shahi:** Data curation; Formal analysis; Investigation; Methodology; **Yongmin Li:** Supervision; **Kiran Shahi:** Writing - original draft preparation; **Yongmin Li:** Writing - review and editing; All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Source code is available at Github and dataset is available at Kaggle. Link for GitHub repository: https://github.com/kiranshahi/Real-time-Background-replacement-in-Video-Conferencing; Link for a data-

set: https://www.kaggle.com/datasets/nikhilroxtomar/person-segmentation.

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

References

1. Beyer, W. Traveling-matte photography and the blue-screen system: A tutorial paper. *J. SMPTE* 1965, *74*, 217–239. doi: 10.5594/J06054.
2. Porter, T.; Duff, T. Compositing digital images. *ACM SIGGRAPH Comput. Graph*. 1984, *18*, 253–259. doi: 10.1145/964965.808606.
3. Rhemann, C.; Rother, C.; Rav-Acha, A.; et al. High resolution matting via interactive trimap segmentation. In *Proceedings of 2008 IEEE Conference on Computer Vision and Pattern Recognition*, *Anchorage*, *USA*, *23–28 June 2008*; IEEE: New York, 2008; pp. 1–8. doi: 10.1109/CVPR.2008.4587441.
4. Boda, J.; Pandya, D. A survey on image matting techniques. In *Proceedings of 2018 International Conference on Communication and Signal Processing* (*ICCSP*), *Chennai*, *India*, *3–5 April 2018*; IEEE: New York, 2018; pp. 765–770. doi: 10.1109/ICCSP.2018.8523834.
5. Xu, N.; Price, B.; Cohen, S.; et al. Deep image matting. In *Proceedings of 2017 IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), *Honolulu*, *USA*, *21–26 July 2017*; IEEE: New York, 2017, pp. 311–320. doi: 10.1109/CVPR.2017.41.
6. Forte, M.; Pitié, F. *F*, *B*, alpha matting. arXiv: 2003.07711, 2020.
7. Sengupta, S.; Jayaram, V.; Curless, B.; et al. Background matting: The world is your green screen. In *Proceedings of 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), *Seattle*, *USA*, *13–19 June 2020*; IEEE: New York, 2020; pp. 2288–2297. doi: 10.1109/CVPR42600.2020.00236.
8. Lin, S.C.; Ryabtsev, A.; Sengupta, S.; et al. Real-time high-resolution background matting. In *Proceedings of 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (*CVPR*), *Nashville*, *USA*, *20–25 June 2021*; IEEE: New York, 2021; pp. 8758–8767. doi: 10.1109/CVPR46437.2021.00865.
9. Zhu, Q.S.; Heng, P.A.; Shao, L.; et al. What's the role of image matting in image segmentation? In *Proceedings of 2013 IEEE International Conference on Robotics and Biomimetics* (*ROBIO*), *Shenzhen*, *China*, *12–14 December 2013*; IEEE: New York, 2013; pp. 1695–1698. doi: 10.1109/ROBIO.2013.6739711.
10. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional networks for biomedical image segmentation. In *Proceedings of the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention*, *Munich*, *Germany*, *5–9 October 2015*; Springer: Berlin/Heidelberg, Germany, 2015; pp. 234–241. doi: 10.1007/978-3-319-24574-4_28.
11. Xie, H.X.; Lin, C.Y.; Zheng, H.; et al. An UNet-based head shoulder segmentation network. In *Proceedings of 2018 IEEE International Conference on Consumer Electronics-Taiwan* (*ICCE-TW*), *Taichung*, *China*, *19–21 May 2018*; IEEE: New York, 2018; pp. 1–2. doi: 10.1109/ICCE-China.2018.8448587.
12. Kuang, Z.J.; Tie, X.R. Flow-based video segmentation for human head and shoulders. arXiv: 2104.09752, 2021.
13. Zhang, Z.X.; Liu, Q.J.; Wang, Y.H. Road extraction by deep residual U-Net. *IEEE Geosci. Remote Sens. Lett*. 2018, *15*, 749–753. doi: 10.1109/LGRS.2018.2802944.
14. Liang, Z.Y.; Guo, K.; Li, X.B.; *et al*. Person foreground segmentation by learning multi-domain networks. *IEEE Trans. Image Process.*, **2022**, *31*: 585−597.
15. Miao, J.; Sun, K.Q.; Liao, X.; *et al*. Human segmentation based on compressed deep convolutional neural network. *IEEE Access*, **2020**, *8*: 167585−167595.
16. Zhang, S.H.; Dong, X.; Li, H.; *et al*. PortraitNet: Real-time portrait segmentation network for mobile device. *Comput. Graph.*, **2019**, *80*: 104−113.
17. Shi, X.J.; Chen, Z.R.; Wang, H.; et al. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Proceedings of the 28th International Conference on Neural Information Processing Systems*, *Montreal Canada*, *7–2 December 2015*; MIT Press: Cambridge, 2015; pp. 802–810.
18. Ballas, N.; Yao, L.; Pal, C.; et al. Delving deeper into convolutional networks for learning video representations. In *Proceedings of the 4th International Conference on Learning Representations*, *San Juan*, *Puerto Rico*, *2–4 May 2016*; 2016.
19. Lin, S.C.; Yang, L.J.; Saleemi, I.; et al. Robust high-resolution video matting with temporal guidance. In *Proceedings of 2022 IEEE/CVF Winter Conference on Applications of Computer Vision* (*WACV*), *Waikoloa*, *USA*, *3–8 January 2022*; IEEE: New York, 2022; pp. 3132–3141. doi: 10.1109/WACV51458.2022.00319.
20. Azad, R.; Asadi-Aghbolaghi, M.; Fathy, M.; et al. Bi-directional convLSTM U-Net with densley connected convolutions. In *Proceedings of 2019 IEEE/CVF International Conference on Computer Vision Workshop* (*ICCVW*), *Seoul*, *Korea* (*South*), *27–28 October 2019*; IEEE: New York, 2019; pp. 406–415. doi: 10.1109/ICCVW.2019.00052.
21. Tomar, N. Person segmentation dataset. 2020. Available online: https://www.kaggle.com/datasets/nikhilroxtomar/personsegmentation (accessed on 10 March 2023).
22. Jordan, J. An overview of semantic image segmentation. 2018. Available online: https://www.jeremyjordan.me/semantic-segmentation/ (accessed on 8 March 2023).
23. Kandeyang, V.K. Image segmentation: Cross-entropy loss vs dice loss. 2020. Available online: https://www.kaggle.com/getting-started/133156/ (accessed on 6 March 2023).

Citation: Shahi, K; Li, Y. Background replacement in video conferencing. *International Journal of Network Dynamics and Intelligence*. 2023, 2(2), 100004. doi: 10.53941/ijndi.2023.100004