3rd International Conference on Industrial Artificial Intelligence (IAI), Nov. 8-11, 2021, Shenyang, China

Development of New Efficient Transposed Convolution Techniques for Flame Segmentation from UAV-captured Images

F M Anim Hossain and Youmin Zhang*
Department of Mechanical, Industrial and Aerospace Engineering
Concordia University
Montreal, Quebec, Canada
fmanim.hossain@concordia.ca, youmin.zhang@concordia.ca

2021 3rd International Conference on Industrial Artificial Intelligence (IAI) | 978-1-6654-3517-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/IAI53119.2021.9619442

***Abstract*—Although Fully Convolutional Networks (FCNs) have been proven to be a very powerful tool in deep learning-based image segmentation, they are still too computationally expensive to be incorporated into mobile platforms such as Unmanned Aerial Vehicles (UAVs) for real-time performance. While significant efforts have been made to make the *encoder* side of a FCN more efficient, the *decoder* side, which involves upsampling the feature maps, is still overlooked in comparison. This paper proposes two new efficient upsampling techniques, "Reversed Depthwise Separable Transposed Convolution (RDSTC)" and "Compression-Expansion Transposed Convolution (CETC)". U-Net architecture and UAV-captured forest pile fire images have been used to evaluate the performance of these new efficient upsampling techniques. RDSTC and CETC achieve Dice scores of 0.8815 and 0.8832 respectively, outperforming commonly used bilinear interpolation and original transposed convolution, while significantly reducing the number of upsampling computations. The results of this paper demonstrate that upsampling operation in a deep learning architecture can be made more efficient without degradation in performance.**

***Index Terms*—efficient transposed convolution, deep learning, unmanned aerial vehicles, forest fire detection, flame segmentation**

I. Introduction

The worlds of machine learning and computer vision were taken by storm when a Convolutional Neural Network (CNN) named AlexNet won the 2012 ImageNet Large Scale Visual Recognition Competition (ILSVRC) in image classification [1]. Since then, CNNs have become a popular tool in computer vision tasks such as image classification, object detection and localization, semantic segmentation, scene understanding, action recognition and even generating new images [2]. Along with the advancements in applications, the architectures of CNNs went through significant evolution and one of the focal points of research has been to make them lightweight and computationally efficient. This is especially important when expecting real-time computer vision operations on a resource-limited mobile platform such as Unmanned Aerial Vehicle (UAV) [3]. One such application of CNNs using UAV as a platform is forest fire detection [4].

Over the past few years, forest fire detection using CNNs has been an active research field and the approaches are illustrated in Fig. 1. In *classification*, the image goes through the CNN and it decides whether there is a fire in the image or not, in *detection*, the flame and/or the smoke in the image is detected and localized and in *segmentation*, the exact pixels of flame and/or smoke are extracted. For a complete forest fire monitoring system, the segmentation task could be the most useful approach as it can help identify the current characteristics, such as size, location etc. and help to understand the future behavior of the fire.

Although UAVs are the ideal platform for autonomous forest fire detection, the modern CNNs, especially networks required for segmentation, are still computationally too demanding to be incorporated into UAVs for real-time and onboard computation [3], [5]. Segmentation tasks require a special type of CNN named Fully Convolutional Networks (FCNs) where the dense connections at the end are removed and replaced by additional convolution operations [6], [7]. Fig. 2 illustrates the idea of FCN-based segmentation through two popular approaches. The first part of a FCN is the same as a typical CNN where the input image gets progressively convolved and turned into (up to 1/32 times) smaller resolution feature maps *encoded* with global semantic information. In the next part, from the *encoded* global features, local information are *decoded* by continuous convolution and upsampling operations. The final prediction of a FCN is a binary mask of the target object. In summary, a FCN has both *encoding* operation (convolution and downsampling) and *decoding* (upsampling) operation, which increases the overall computational complexity and memory requirements of the architecture. Over the years, significant research has been done on making the *encoding* operation computationally efficient through innovative techniques such as separable convolution, depthwise separable convolution, group convolution etc. [8]–[11]. However, only a handful of researches have focused on the *decoding* (upsampling) operation more efficient and therefore, FCN still remains a computationally complex process ill-suited for UAV operations [12].

The focus of this paper is to develop efficient upsampling techniques to replace the commonly used bilinear interpolation and transposed convolution and reduce the computational com-

Fig. 1. Forest fire detection approaches using a CNN: left) classification, center) flame/smoke detection and right) flame/smoke segmentation

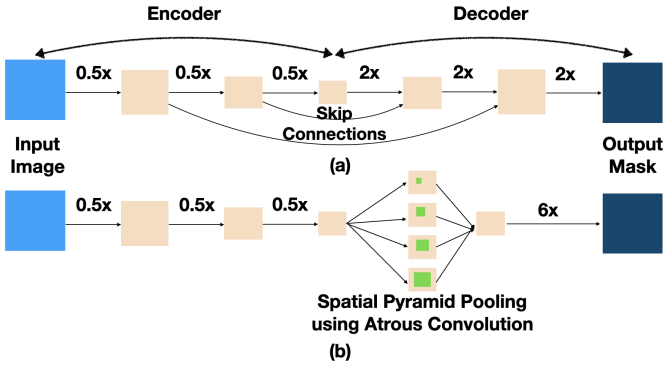

Fig. 2. Two approaches of semantic segmentation with FCNs: a) encoder-decoder and b) spatial pyramid pooling

plexity of FCNs. Here, two efficient upsampling techniques, "Reversed Depthwise Separable Transposed Convolution (RDSTC)" and "Compression-Expansion Transposed Convolution (CETC)" have been proposed. A popular FCN named U-Net [13] has been used as the baseline in conjunction with the latest FLAME dataset [14] that consists of UAV-captured pile burn images. The performance of RDSTC and CETC was evaluated by replacing the upsampling layers of U-Net. The organization of this paper is as follows: literature relevant to this research has been briefly highlighted in Section II, the principal methodologies behind the development of the efficient upsampling methods have been elaborated in Section III and the results have been reported in Section IV. The expected outcome of the proposed method is that the new efficient upsampling techniques will require fewer computations compared to the original transposed convolution without degradation in performance.

II. Existing Decoding Methods

In FCNs, a *decoding* layer attempts to reverse the effects of convolution operations during *encoding* and retrieve the original image information. This typically involves at least one upsampling operation which is, optionally, preceded and/or succeeded by a convolution operation. The scope of this paper is the upsampling operation, which can be done by a non-learnable operation such as bilinear interpolation or learnable operation such as transposed convolution.

One of the first works of upsampling a feature map to retrieve original images could be traced back to Zeiler and Fergus [15], who tried to visualize the intermediate layers of a CNN to understand how it learns to classify different objects. They used transposed convolution for upsampling the intermediate layers and currently it is the most popular upsampling method in *decoding* layers [12], [16]. Wojna *et al*. [12] investigated modifications of transposed convolution by implementing well-known efficient convolution methods such as decomposed convolution [10] and depthwise separable convolution [8]. However, these modifications did not make the transposed convolution operation more efficient.

Among non-learnable methods of upsampling, bilinear and bicubic interpolation methods are popular [17]. However, these "fixed-filter" methods have been found to generate coarse images and they are needlessly computationally heavy [12], [18]. To overcome this issue, Shi *et al*. [18] implemented an innovative method called subpixel convolution. In this method, the original input image is downsampled by a factor of r through consecutive convolution operations and the final *encoded* layer has r^2 feature maps. The feature maps of the final layer are then rearranged to form a single feature matrix of the same resolution as the input image. According to Shi *et al*. [18], this method is more efficient than bilinear and bicubic upsampling and generated finer output. However, Wojna *et al.* [12] asserted that this method has a drawback of introducing alignment artifact.

Till date, the most comprehensive study on *decoders* have been done by Wojna *et al.* [12]. They investigated all the upsampling algorithms addressed in this section along with proposing a new and efficient non-learnable upsampling method called "bilinear additive upsampling", which performed strongly in comparison to the state-of-the-art methods. Their study truly highlighted the importance of upsampling algorithms in CNN operation by demonstrating that a network's performance varies significantly with the change of how the feature maps are upsampled.

It is apparent from this literature review that despite being an important operation of a fully convolutional network, *decoding* algorithms have not received enough focus. This paper builds on this idea and proposes two new efficient upsampling methods, which are described in the following section.

III. Methodology

Both new efficient upsampling techniques proposed in this paper are based on transposed convolution. Therefore, the general operation of transposed convolution has been briefly explained first. Furthermore, RDSTC modifies the idea of depthwise separable convolution into upsampling operation

and therefore, a subsection has been dedicated to elaborate this idea for clarification.

A. Tranposed Convolution

Transposed convolution is often referred to as "deconvolution" or "backward strided convolution" and the operation is illustrated in Fig. 3. In this operation, each element of the input feature map is multiplied with each element of the kernel. The spacing of the results of these multiplications is determined according to the stride of the operation. For example, the stride is 1 in Fig. 3 and therefore, the outputs of the individual multiplications are placed 1 row/column apart from each other. If the stride was set to 2, there would have been a gap of 2 rows/columns between consecutive multiplication results. In the end, the results of the multiplications are summed, forming a feature map with a higher resolution than the input. This method has been used for upsampling in semantic segmentation, instance segmentation, image reconstruction, depth prediction etc. [12].

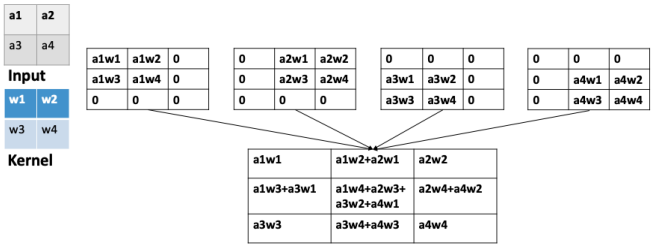

Fig. 3. Tranposed convolution

B. Depthwise Separable Convolution

Depthwise separable convolution [8], [9] is an effective tool in reducing the computational complexity of a CNN and the process is illustrated in Fig. III-B. In a typical convolution operation (Fig. III-B(a)), if the number of input channels (also known as feature maps) is n, a set of n-channel kernel is required to create one output channel. Each kernel channel performs a convolution operation on the corresponding input channel and then sums up to create the output channel. If y-channel output is desired, then this process is repeated y times with y sets of n-channel kernels. Therefore, the total number of operations in a typical convolution operation is output width $\times$ output height $\times$ kernel width $\times$ kernel height $\times$ input channel $\times$ output channel. On the other hand, Fig. III-B(b) illustrates the depthwise separable convolution operation. The overall operation is done in two steps: 1) depthwise operation where only 1 kernel channel is needed to create 1 output feature map and 2) a 1×1 pointwise convolution to change the depth into the desired number of feature maps. This process requires output height $\times$ output width $\times$ input channel $\times$ (kernel width $\times$ kernel height + output channel) computations.

For example, if a $16\times16\times3$ input feature map needs to be turned into a $16\times16\times64$ output feature map using 3×3 kernels, it would require 442,368 operations using conventional convolution and just 56,064 operations using depthwise separable convolution. That is a reduction of almost 87%. This idea is the inspiration behind the proposed Method 1 in this paper.

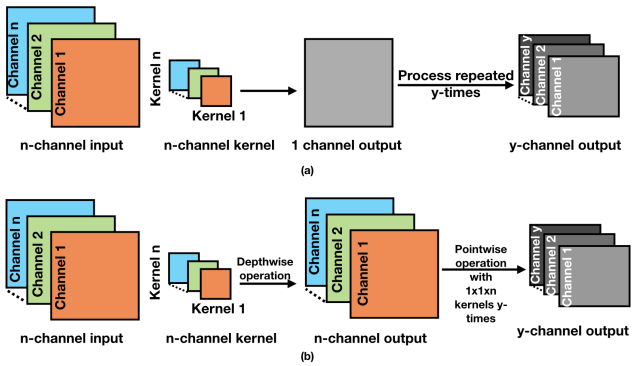

Fig. 4. a) Normal convolution operation, and b) depthwise separable convolution

C. Proposed Method 1: Reversed Depthwise Separable Transposed Convolution

The first efficient upsampling method proposed in this paper is inspired by depthwise separable convolution. Recently, depthwise separable convolution system has been adopted into depthwise separable transposed convolution where a depthwise separable transposed convolution was followed by a pointwise convolution [19], [20]. However, this direct adaptation requires (input height $\times$ input width $\times$ input channel $\times$ kernel height $\times$ kernel width) + (output height $\times$ output width $\times$ input channel $\times$ output channel) number of operations, which ends up making more computations than the straightforward transposed convolution.

In the proposed Method 1, as illustrated in Fig. 5(b), the depthwise separable convolution has been modified to "reversed" depthwise separable transposed convolution by performing the pointwise convolution first and then applying depthwise separable transposed convolution to expand its resolution. This reversal results in input height $\times$ input width $\times$ output channel $\times$ (input channel + kernel height $\times$ kernel width) number of operations.

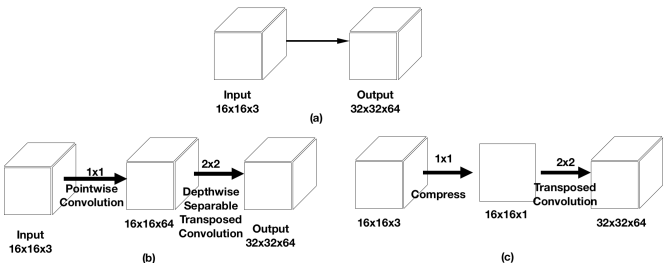

Fig. 5. a) A sample upsampling task, b) Reversed Depthwise Separable Transposed Convolution, and c) Compression-Expansion Transposed Convolution

D. Proposed Method 2: Compression-Expansion Transposed Convolution

Fig. 5(c) illustrates the proposed Method 2, CETC, where a 1×1 pointwise convolution compresses the input feature maps into just one feature map and then performs transposed convolution on it to reach the desired output resolution. This version requires input height $\times$ input width $\times$ (input channel + output channel $\times$ kernel height $\times$ kernel width) number of operations.

Table I expands the list of different decoding methods and their mathematical formula reported in [12], by adding the two proposed methods in it. To highlight the efficiency of the proposed methods, the list includes the number of computations required to perform the upsampling task of Fig. 5(a). 3×3 kernels were assumed for convolution operations whereas 2×2 kernels were assumed for transposed convolution operations. It could be observed that both proposed methods decrease the computational requirements than the existing methods. For the upsampling operation illustrated in Fig. 5(a), RDSTC and CETC reduce the computation of original transposed convolution by 41.66% and 66.27%, respectively.

TABLE I
COMPARISON BETWEEN DIFFERENT UPSAMPLING METHODS

Upsampling Method	Formula	Number of Operations in Fig.5a	Notations
Transposed Convolution	whWHIO	1,96,608	W = input width
Decomposed Transposed	(w+h)WHIO	1,96,608	H = Input height
Conv+Depth-to-Space	whWHI(4O)	7,86,432	I = Input channels
Bilinear Additive Upsampling	wh(2W)(2H)(I/4)O	1,96,608	O = Output channels
Proposed Method 1- RDSTC	WHO(wh+I)	1,14,688	w = Kernel width
Proposed Method-2 CETC	WH(I+whO)	66,304	h= Kernel height

IV. RESULTS

The performance of the proposed efficient upsampling methods has been evaluated using the popular U-Net architecture and the newly developed FLAME dataset [14]. The upsampling operations in the decoding layers of U-Net have been replaced by bilinear upsampling, original transposed convolution, RDSTC and CETC. $128\times128\times3$ sized images were used to train the networks with ADAM optimizer, batch size of 32 and for 40 epochs. The networks were trained on a desktop computer with Intel Core-i9, 16GB RAM, and 10GB NVIDIA RTX 3080 Graphical Processing Unit (GPU).

A. Dataset

The FLAME dataset used for the evaluation of the proposed upsampling techniques is developed by Shamsoshoara *et al.* [14] and contains about 2000 frames of annotated UAV-captured aerial images of pile burns. From this dataset, 1440 images were used to train and 560 were used to evaluate the networks. The image background consists of forestry and snow and the original resolution of the images is 3840×2160. The dataset consists of both large and minuscule flame regions, which makes it ideal to train a network to detect flame of different sizes.

B. Network Architecture

As addressed previously, the U-Net architecture, developed by Ronneberger *et al.* [13], is used in this paper as the baseline architecture due to its popularity in segmentation tasks. The structure of U-Net is similar to Fig. 2(a) and in the original paper, transposed convolution was used for upsampling. In this paper, 4 variants of the U-Net have been used by modifying the upsampling operation with bilinear interpolation, RDSTC and CETC. The input to the network has a resolution of $128\times128\times3$ and during the *encoding* operation, each block consists of two successive convolutions to double the depth of the feature maps followed by a max-pooling operation to halve the input resolution. After four such layers, the final output of the *encoder* feature maps ends up at $8\times8\times512$. In the *decoding* side, an upsampling operation doubles the input resolution while halving the depth of the feature maps and then concatenates with the corresponding *encoding* layer. This is followed by two convolutions and another upsampling, continuing until the feature maps get to the resolution of the original input. Finally, a softmax operation determines the final prediction.

C. Performance Metric

The performance of the networks was evaluated using Dice score, a popular evaluation metric for the purpose of semantic segmentation and it is mathematically described using (1), where y_{pred} is the prediction made by the network and y_{true} is the ground truth [21]. Dice coefficient can range between 0 and 1 and a score of 1 means a perfect overlap between the prediction and ground truth.

$$Dice\ Score = \frac{2\sum_{pixels} y_{true} * y_{pred}}{\sum y_{true}^2 + \sum y_{pred}^2} \tag{1}$$

D. Quantitative Analysis of the Evaluation Results

Table II displays the performance comparison of the different upsampling methods with U-Net as the base architecture. The accuracies represented in Dice score show that the performances have been similar but the proposed RDSTC and CETC upsampling techniques outperformed both bilinear interpolation and straightforward transposed convolution. Furthermore, both RDSTC and CETC reduced the upsampling computation requirements from bilinear interpolation and transposed convolution significantly. RDSTC has almost 97% and 74% fewer computations than bilinear interpolation and transposed convolution respectively whereas CETC reduces the upsampling computation by almost 99% and 98.9% compared to bilinear interpolation and transposed convolution respectively. The number of parameters required for RDSTC and CETC is fewer than the other two common methods as well, leading to fewer memory requirements, which is important when designing a network for UAV application. Although the runtime performance was similar in the GPU, using the CPU it could be observed that both RDSTC and CETC run faster than the other two methods, which is also

a significant parameter to consider when UAV application is involved.

TABLE II
PERFORMANCE COMPARISON

Method	Total Upsampling Computations	Total Parameters	CPU processing time (sec)	Dice Score
Bilinear interpolation	1.21E+09	8,637,345	0.041	0.8784
Transposed convolution	1.34E+08	8,550,785	0.039	0.88
RDSTC	3.45E+07	8,030,465	0.037	0.8815
CETC	1.47E+06	7,857,349	0.036	0.8832

E. Qualitative Analysis of the Evaluation Results

Qualitatively, the resulting segmentations were similar for all four methods as demonstrated in Fig. 6. The validation dataset consisted of challenging images with minuscule flame pixels, some of them difficult for even human eyes to detect. The sample results of Fig. 6 demonstrate the strength of deep learning networks for fire segmentation tasks.

The top row in Fig. 6 consists of six burning piles, with four of them partially hidden behind trees. Bilinear interpolation and CETC both detected all six burn spots while transposed convolution and RDSTC detected five of them. The middle row is an example of the bilinear interpolation method detecting false flame pixels, demonstrating that its sensitivity is high, allowing more false positives to be detected. This higher rate of false positives possibly resulted in the slightly lower Dice score of bilinear interpolation than the other three methods. The bottom row is another challenging image with seven flame regions where all four methods demonstrated strong performances by detecting all of them. The qualitative results show that while all four methods demonstrated strong performance, the performances of RDSTC and CETC are exceptional given the significant reduction in upsampling computations compared to the other two methods. The qualitative results conclusively assert the strength of fully convolutional networks for a challenging task such as flame segmentation from UAV-captured images.

CONCLUSION AND FUTURE WORKS

Performances of the two new efficient upsampling techniques proposed in this paper exceeded expectations when evaluated with UAV-capture aerial pile burn images in the benchmark FLAME dataset. The results of this paper have significance in two aspects: firstly, despite using fewer parameters and significantly fewer upsampling computations than popular bilinear interpolation and transposed convolution techniques, the newly proposed methods performed strongly. Secondly, the results of this paper demonstrated the strength of deep learning techniques to detect and segment UAV-captured minuscule flame pixels, which is a challenging task. These efficient transposed convolution techniques can also be incorporated into other networks that require upsampling operations such as instance segmentation, depth estimation, image reconstruction etc.

The future works will involve expanding the dataset with aerial images that contain more varying backgrounds, such as fall leaves, setting sun, smoke and fog etc. Furthermore, the segmentation task will be expanded to segment both flame and smoke. The final objective is to be able to create a lightweight deep learning architecture that can be incorporated onboard a UAV for real-time segmentation. This will involve finding an optimum architecture with innovative *encoding* and *decoding* methods and evaluating them through rigorous simulation, laboratory and field experiments. The results of this paper set the first steps to achieving this overall objective.

V. ACKNOWLEGEMENTS

This work was financially supported in part by Natural Sciences and Engineering Research Council of Canada.

REFERENCES

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in Neural Information Processing Systems*, vol. 25, pp. 1097–1105, 2012.

[2] S. Khan, H. Rahmani, S. A. A. Shah, and M. Bennamoun, "A guide to convolutional neural networks for computer vision," *Synthesis Lectures on Computer Vision*, vol. 8, no. 1, pp. 1–207, 2018.

[3] A. Carrio, C. Sampedro, A. Rodriguez-Ramos, and P. Campoy, "A review of deep learning methods and applications for unmanned aerial vehicles," *Journal of Sensors*, vol. 2017, 2017.

[4] M. A. Akhloufi, N. A. Castro, and A. Couturier, "UAVs for wildland fires," in *Autonomous Systems: Sensors, Vehicles, Security, and the Internet of Everything*, vol. 10643, p. 106430M, International Society for Optics and Photonics, 2018.

[5] F. M. A. Hossain, Y. M. Zhang, and M. A. Tonima, "Forest fire flame and smoke detection from UAV-captured images using fire-specific color features and multi-color space local binary pattern," *Journal of Unmanned Vehicle Systems*, vol. 8, no. 4, pp. 285–309, 2020.

[6] L. C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 801–818, 2018.

[7] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[8] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1251–1258, 2017.

[9] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "Mobilenets: Efficient convolutional neural networks for mobile vision applications," *arXiv:1704.04861*, 2017.

[10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

[11] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1492–1500, 2017.

[12] Z. Wojna, V. Ferrari, S. Guadarrama, N. Silberman, L. C. Chen, A. Fathi, and J. Uijlings, "The devil is in the decoder," in *British Machine Vision Conference 2017*, pp. 1–13, BMVA Press, 2017.

[13] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-assisted Intervention*, pp. 234–241, Springer, 2015.

[14] A. Shamsoshoara, F. Afghah, A. Razi, L. Zheng, P. Z. Fulé, and E. Blasch, "Aerial imagery pile burn detection using deep learning: the flame dataset," *Computer Networks*, p. 108001, 2021.

[15] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *European Conference on Computer Vision*, pp. 818–833, Springer, 2014.

[16] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv:1603.07285*, 2016.

[17] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

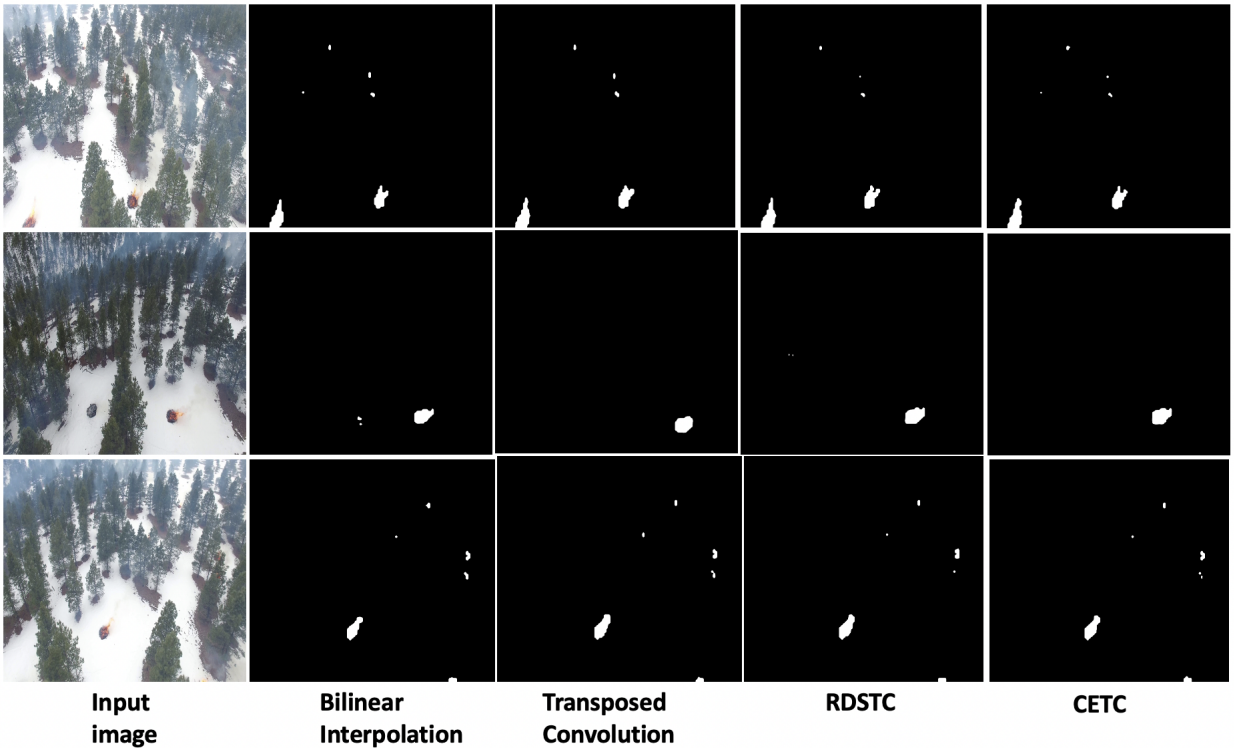

Fig. 6. Sample flame segmentation results

[18] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1874–1883, 2016.

[19] F. Zhong, M. Li, K. Zhang, J. Hu, and L. Liu, "DSPNet: A low computational-cost network for human pose estimation," *Neurocomputing*, vol. 423, pp. 327–335, 2021.

[20] L. Bai, Y. Zhao, M. Elhousni, and X. Huang, "DepthNet: Real-time LiDAR point cloud depth completion for autonomous vehicles," *IEEE Access*, 2020.

[21] T. Eelbode, J. Bertels, M. Berman, D. Vandermeulen, F. Maes, R. Bisschops, and M. B. Blaschko, "Optimization for medical image segmentation: Theory and practice when evaluating with dice score or jaccard index," *IEEE Transactions on Medical Imaging*, vol. 39, no. 11, pp. 3679–3690, 2020.