A Lightweight Self Attention Based Multi-Task Deep Learning Model for Industrial Solar Panel and Environmental Monitoring

Tuhinangshu Gangopadhyay, Tanushree Meena, Debojyoti Pal, and Sudipta Roy

***Abstract*—Environmental monitoring has become a serious topic of discussion and is gaining mass attention. The reason is the severe consequences of environmental depletion, which has led to circumstances like climate change, rise in floods and droughts, changed rainfall patterns, etc. So, various measures are being taken to protect the environment, like shifting to renewable and pollution-free energy alternatives, like solar energy, and handling the after-effects of disasters, like flood management and oil spill accident management. However, their identification still remains a huge challenge, which is laborious and extensive. Thus, this work proposed a lightweight and efficient segmentation model, SA U-Net++, for the automatic identification of solar panels and their associated defects, flood affected-areas and oil spill accident regions. The model's novel blend of level-wise self-attention modules is embedded with the revised bridge connections and the dropouts. It has helped in better efficient global context understanding and feature extraction from the inputs, besides maintaining the integrity of the training process and avoiding some major learning and run-time issues, like overfitting and memory exhaustion. Our detailed experiments demonstrate that the proposed model outperforms state-of-the-art models. The results confirm its high generalizability, cost-effectiveness, and robustness.**

***Index Terms*—Flood segmentation, oil spill segmentation, self-attention, semantic segmentation, solar panel segmentation and defect identification.**

I. Introduction

ENVIRONMENTAL Degradation has grown out to be one of the biggest threats to the existence of life on Earth [1][2]. With an ever-expanding population, the planet is facing tremendous environmental issues every passing day. These manifest in various ways, including pollution, deforestation, release of dangerous gases, and global warming, only to name a few. The rapid rise in the environmental temperature and irregular rainfall patterns leading to droughts and melting down of the snow icecaps, which is accompanied by an increase in sea level and flood frequency, and the extinction of species due to changing environmental conditions are just some of the consequences of environmental depletion [3]. Therefore, environmental monitoring and protection is gaining importance from the various governments and organisations around the globe. A major initiative towards environmental conservation can comprise of tackling the environmental pollution problem. The replacement of fossil fuels with renewable sources of energy, like solar energy is a well-recognized solution. Fossil fuels are known for their ill effects like air pollution, climate change and global warming [4]. Further, these fossil fuels are exhaustive in nature. A study has shown that the amount of oil, coal and gas left in this world can only be used for 35 years, 107 years and 37 years respectively [5]. Another widely known pollution threat to the marine environment is the leaking of oil from transport ships and tankers. Oil, being lighter than water, floats on ocean surfaces, forming a barrier between air and water and preventing the exchange of oxygen. This can cause massive damage to marine life and environment [7] and it should be prevented at any cost. An equal focus should also be given to the after-effects of any natural calamity, like flooding. Floods not only endanger human lives, but the dirty water can also negatively impact the soil, land as well as the flora and fauna of any habitat [6].

Solar panels are developed to harness solar energy in various parts of the world. They have the potential to deal with the rising energy demands, owing to factors like the rapid increase in population, setting up of new establishments and upgrading the lives of mankind.

However, these solar panels need regular inspection and examination in search of underlying defects (that can arise due to their exposition to harsh environmental conditions and can affect their efficiency). These tasks are highly challenging, time-consuming and subjected to human inaccuracies.

Similarly, searching for flood-affected areas and oil spill accidents in large geographical areas is an extensive task and requires both time and effort. Thus, there is a serious need for automation solution in this field, so as to tackle these problems efficiently, in real-time and without any scope for errors.

The rapid improvement in Artificial Intelligence (AI) and Deep Learning (DL) techniques, accompanied by enhancements in GPU and hardware technologies, can help deal with the above-mentioned environmental issues. A model incorporated with the satellite/aerial imagery system can help in the real-time

Manuscript received 20 May 2024; accepted 20 July 2024. *(Corresponding author: Sudipta Roy.)*

The authors are with Artificial Intelligence and Data Science, Jio Institute, Navi Mumbai, India (e-mail: tuhinangshu.ganguly@gmail.com; tanushree.meena@jioinstitute.edu.in; mail13dp.99@gmail.com; sudipta1.roy@jioinstitute.edu.in).

The code is uploaded in: https://github.com/tg2001/SA-UNetpp.git

Recommended for acceptance by J. Ding.
Digital Object Identifier 10.1109/TETCI.2024.3444590

identification of solar panels (along with their underlying defects), floods and oil spills.

However, DL models suffer from certain drawbacks. One such problem is the generalisability issue. A non-generalisable model will perform well for only one task and is indicative of the overfitting nature of the model. Another problem regarding DL is its scalability. A deeper model generally performs better than a lighter one, accompanied by an exponential increase in the number of parameters, leading to an increased model complexity, time, resources and data for training and testing purposes. This may pose a problem in scenarios with less data availability, or during deployment where the model is restrained by time (self-driving cars) or resources (low-end edge devices).

Consequently, an automatic and reliable technique needs to be in place to address these concerning issues effectively. Thus in this work, we have proposed a novel architecture, SA U-Net++, for addressing these issues. This model represents a pioneering effort aimed at addressing these challenges, which can later be easily extended to solve other problems like identification of illegal deforestation from aerial imageries, for example. Prior literature addressing similar issues has been inadequate in giving an appropriate solution, lacking thorough critical assessments of their proposed solutions, as detailed in the following section on Related Works. SA U-Net++ put forward a novel combination of level-wise self-attention modules, embedded with the redesigned cross connections and dropouts in inner levels. This fusion assists in better feature extraction and representation and global-context understanding in the hidden layers, alongside alleviating the scalability and overfitting issues of the model, thus maintaining robustness during training. The following contributions are made to this work:

1) We have proposed a novel DL model, named SA U-Net++, for the automatic segmentation of four different types of images for four different tasks: solar panels and their corresponding defects, floods and oil-spills.
2) The proposed model has good scalability, i.e., it uses fewer variables (thus being lightweight) and has an optimal combination of parametricity and inference time (Fig. 10). Further, the use of self-attention uplifts scalability, which provides enhanced feature extraction with fewer parameters.
3) The model mitigates overfitting on the dataset with the use of dropout layers in the deeper levels of the model. The use of the dropouts in our model has also been experimentally justified, as discussed later.
4) Our model has achieved generalizability to a great extent. The model's performance on multiple different tasks elucidates this. It has the capability of learning complex tasks with less data, besides being reproducible.

Various experimentation procedures testify to the efficiency and performance improvement yielded by our model over the state-of-the-art models.

II. Related Works

DL-based segmentation techniques are not new to the market. Some initial segmentation architectures included the 3D convolutional neural network (CNN) model [8], and fully connected network (FCN) [9]. However, these models suffer from the high-level information gap between the encoder and decoder, which contains the orientational data.

A breakthrough was seen with the introduction of the U-Net architecture by Ronneberger et al. [10]. It used the encoder-decoder based architecture, with additional skip connections between the encoders and decoders. An encoder downsamples the model input and retrieves essential semantic data, followed by the decoder upsampling the same. It has shown a significant performance improvement. In 2018, U-Net++ [11] was proposed, which rearranged the skip connections, to combine the multi-level encoder data with the decoder data at each step for easier and better segmentation. In 2020, Huang et al. suggested that both U-Net and U-Net++ fail to take advantage of the information from full-scale input images, and proposed U-Net 3+ [12] to tackle this. In the decoding stage, U-Net 3+ draws connections from the lower-level decoders and the higher-level encoders to combine them for better segmentation. However, this model uses numerous layers for combining different level semantic data in the model, thus making it highly parameterised. An attention-based U-Net architecture [13] was proposed in 2018, which uses attention gates to enhance the skip connection output. In the same year, Alom et al. proposed R2U-Net [14], which used residual as well as recurrent networks in the encoder and decoder blocks. However, using several parallel residual and recurrent connections in deep segmentation models at multiple levels can make the model quite complicated, resulting in overfitting and increased time complexity.

Some other segmentation architectures include DeepLabv3+ [15], which uses specialised modules like Atrous Convolution (AC) and Atrous Spatial Pyramid Pooling (ASPP) for capturing multi-scale data from the image; LinkNet [16], which utilized the skip connections of U-Net, with the addition operation instead of concatenation for combining the skip connection and decoder outputs; SegNet [17], which uses a series of convolutional layers for performing segmentation; and RFBSNet [18], which has an architecture similar to the U-Net model, with an additional sub-branch from the downsampling module. DeepLab structure misses out on certain data in trying to expand the field of view, thus increasing the chances of missing out on essential information. The use of the addition operation in Linknet, however, could not bring large difference in performance. SegNet has low parametricity, which lacks skip connections, thus making it suitable mostly for basic tasks. And finally, RFBSNet also suffers from the information gap between the encoder and decoder, thus the performance improvement was low.

A. Solar Panel Segmentation

To this date, not much work on solar panel segmentation is known to exist. Zhuang et al. [19] trained several U-Net models simultaneously, with the models improving each other's performance through cross-learning, for the segmentation of residential solar panels. However, this method is highly time, space and data-intensive as it involves training and testing multiple U-Net models. Zhu et al. [20] presented a deep learning network, named Deep solar PV refiner, for the enhancement of solar PV segmentation using a Split-Attention Network, Dual-Attention

Network and Atrous Spatial Pyramid Pooling. Wani et al. [21] proposed a U-Net based model, with a Mobilenet encoder, for solar panel segmentation in 2021. Camilo et al. [22] used a SegNet-based model for this task. The SegNet model used three pairs of encoding and decoding blocks, with an input size of 41 × 41. The input size considered here is quite small, not enough to capture the minimum required scenario, and a proper performance evaluation of the model is also missing. Parhar et al. [23] proposed to perform solar panel classification using EfficientNet-B7 model, followed by segmentation using U-Net with an EfficientNet backbone for identification of solar panel locations and surface area. However, the use of classification backbones for segmentation makes the model heavy, and intensive and also introduces problems like vanishing and exploding gradient problems in very deep backbones. Also, using enhancement modules on top of segmentation output increases the overall prediction time.

B. Solar Panel Defect Identification

Previous works on solar panel defect identification mostly have been a classification or detection task. Chen et al. [24] proposed a CNN model structure to classify a defect on a solar panel using its multi-spectral properties. A faster RCNN model, with a ResNet50 backbone, was suggested in [25] for solar panel defect detection from thermal imageries. Balcioglu et al. [26] suggested a deep CNN-based architecture for the classification of surface defects on solar panels, using their distinguishable colour characteristics. No work on solar panel defect segmentation is known to exist to date. Segmentation is, however, preferred over classification and detection as it gives a complete overview of the type, location, shape and extent of the defect, along with its possible effect on the consecutive cells.

C. Flood Segmentation

Recently, flood segmentation has drawn the attention of researchers. Hernandes et al. proposed an AI-based technique [27] for flood segmentation using three segmentation models, with three different backbones; and presented a detailed comparison report of the same. Rahnemoonfar et al. [28] provided a publicly available flood segmentation dataset, named FloodNet, for enabling further research in this field. The paper also reported the performance of three classification models and three segmentation models on this dataset. As can be seen, no proper work dedicated to flood segmentation actually exists, except for some comparison studies on the existing methods.

D. Oil Spill Segmentation

Previous works on oil spill segmentation include a DL based model proposal by Yekeen et al. [29] based on Mask RCNN, that used ResNet101 as the backbone with FPN for feature extraction. Krestenitis et al. proposed an oil spill segmentation dataset with five classes [30] for assisting research in this domain. They further trained a DeepLab segmentation model [31], equipped with a ResNet101 model, that uses AC and ASPP for capturing scenarios with different field-of-views. However, using AC and ASPP eliminates a large portion of the data in its field of view, eventually increasing the chances of missing out on essential information. Also, as mentioned earlier, using backbones in segmentation architectures can make the model heavy, intensive and prone to gradient problems.

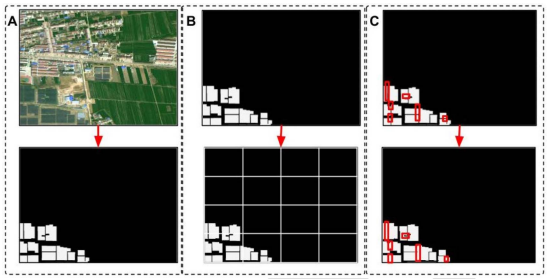

Fig. 1. (a) A satellite-captured 1024 × 1024 image and its corresponding segmentation, (b) Division into 256 × 256 patches, (c) Interpolation to size 256 × 256 (Red boxes indicate the differences in interpolation input and output).

III. Methodology

A few preprocessing steps have been followed on the raw dataset for each task before obtaining the final dataset. This final dataset is fed into the proposed model, followed by different tests and experimentations.

A. Dataset Availability

Solar Panel Segmentation (Task T1): The multi-resolution photovoltaic panel dataset [32] has been used for this task. It contains 3716 satellite-captured images of size 256 × 256 and 1024 × 1024, with segmentation into two classes: Solar panel and background pixel.

Solar Panel Defect Segmentation (Task T2): This task uses the solar panel defect dataset from MIAD [35]. It contains 5000 UAV images of size 512 × 512, with segmentation into four classes: Good cells, broken cells, presence of a foreign body and missing cells.

Flood Segmentation (Task T3): The FloodNet dataset [28] is used for this task. It contains 2343 UAS images of size 3000 × 4000, segmentated into 10 classes: Background, Building Flooded, Building Non-Flooded, Road Flooded, Road Non-Flooded, Water, Tree, Vehicle, Pool and Grass.

Oil Spill Segmentation (Task T4): The oil spill dataset, provided as a part of the ROBORDER project [30], has been used for this task. It contains 1110 satellite-captured images of size 650 × 1250, with segmentation into 5 classes: Oil spill, look-alike, land, ship and sea areas.

B. Data Preprocessing

The photovoltaic panel dataset [32] contains images of two sizes: 256 × 256 and 1056 × 1056 (Fig. 1(a) shows an input sample). The first step is to resize all the images to the same dimension to give them as input to the model. Using a

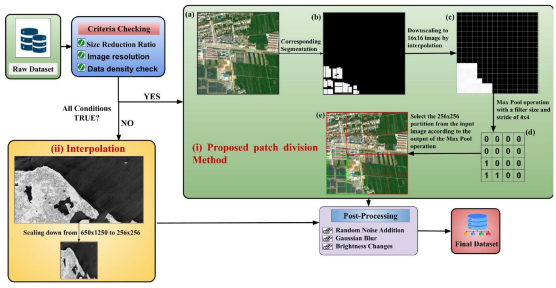

Fig. 2. Data Pre-Processing. (I) Process of division into patches, followed by the selection of the appropriate ones for inclusion in the dataset. (a) The input image, (b) Corresponding segmentation, (c) Segmentation downsampled to size 16x16, (d) Maxpool output, and (e) The green/red boxes indicate the patches to be selected/rejected as per the maxpool output.

Fig. 3. The input data samples and their corresponding patches of (a) Solar Panel defect dataset, (b) Flood Segmentation dataset, and (c) Oil Spill dataset.

1024 × 1024 input size would make the model highly parameterized and computationally expensive. Thus 256 × 256 images are preferable as input. Two ways to scale down the images:

1) Using interpolation to downscale the images results in certain imperfections in the segmentation output (Fig. 1(c)). So it should be avoided wherever possible.
2) Dividing the large image into sixteen 256 × 256 patches (Fig. 1(b)). This would preserve the segmentation output. But this is feasible only if each patch has enough contextual data for successful segmentation.

The second method is preferred. In this case, not all patches can be kept in the dataset, otherwise, it would become too large. Some of them need to be discarded. Random selection of patches may result in choosing the ones not having any solar panels (Fig. 1(b)).

A simple method is devised to ensure the collection of only the necessary 256 × 256 patches Fig. 2(i). The maxpool output will have 1 for the patches having solar panels. The proposed method is preferred over checking for the presence of the solar panel class in any patch because our method ensures the collection of patches having a significant number of solar panels, which will be valuable to our training process (otherwise, it could lead to some unwanted biases during training due to data imbalance). The final dataset for Task T1 has 6300 images used for training and the remaining 500 for testing.

The general algorithm for the proposed simple preprocessing is described in Algorithm 1. This general algorithm accepts an image and its corresponding segmentation of any dimension $M \times N$, along with the required dimension of the output patches $A \times B$. The maxpool stride size is the same as the filter size to avoid data overlapping and reduce time complexity and is set to 4×4 by default. The output dimensions should be in the power of 2 to avoid any unnecessary image border removal

Algorithm 1: Proposed Patch Division Method.

Require: Input image I and corresponding segmentation S_I with dimensions $M \times N$
Require: Required output dimension of each patch $A \times B$
Ensure: Output the required set of patches S

$S = []$
$K = 4$
$L = 4$ {$K \times L$ is the maxpool filter size}
$X = \lfloor \frac{M}{A} \rfloor$
$Y = \lfloor \frac{M}{B} \rfloor$ {X and Y store the maximum possible number of patches along each dimension}
$P = \lfloor \frac{A}{K} \rfloor$
$Q = \lfloor \frac{B}{L} \rfloor$ {$P \times Q$ is the dimension of the interpolation output}
$I\_ = INTERPOLATION(S\_I, P, Q)$
$J = MAXPOOL(I\_, K, L)$
for $x = 0$ **to** $X - 1$ **do**
  for $y = 0$ **to** $Y - 1$ **do**
    if $J[x, y] = 1$ **then**
      $x\_ = x \times A$
      $y\_ = y \times B$
      Add $I[x\_ : x\_ + A, y\_ : y\_ + B]$ to S
    end if
  end for
end for

issues at the maxpool stage. To resolve this issue (in case the output dimensions are non-exponent values of 2), the maxpool stride needs to be adjusted. For our use case, the input image size is 1024 × 1024 and the required output size is 256 × 256. The patching method, however, could not be applied to the other three tasks, as the resulting images would lack adequate background information and contextual data. For Task T2, the division into patches just zooms into the defect area without much difference in surrounding data, so any process can be applied here (Fig. 3(a) illustrates the same). But, for Tasks T3 and T4, Fig. 3(b) and 3(c) respectively represent a significant loss in neighbouring data, which can lead to some unexpected outputs. The application of the patching process depends on the size reduction ratio, the resolution and the information density of the dataset images. The data density, ρ indicates the amount of necessary information contained in each image and is calculated as the ratio of the number of essential pixels to the total number of pixels in the image, as shown below:

$$\rho = \frac{p}{M \times N} \tag{1}$$

Where ρ represents the data density of the image, p represents the task essential pixel count and M and N represent the dimension of the input image data. The data density of the whole dataset can be expressed using probability density estimation with a Gaussian kernel (Fig. 4), and is computed as follows:

$$P(\hat{\rho}) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{\hat{\rho} - \rho_i}{h}\right) \tag{2}$$

Here $P(\cdot)$ represents the probability density of a function at any given point $\hat{\rho}$ with a Kernel function $K(\cdot)$; n represents the

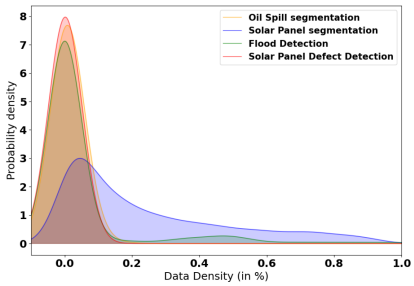

Fig. 4. The data density comparison of all the datasets. This image demonstrates that the overall data density of datasets for solar panel defect segmentation, flood segmentation and oil spill segmentation is not high enough for patching operations to be performed.

number of data points/images, ρ_i represents the data density of ith image and h represents the bandwidth. The Gaussian Kernel, $K(x)$, is defined as:

$$K(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} \tag{3}$$

A high ρ value is expected from the larger images so that a significant amount of data can be present in the smaller patches. Fig. 4 shows that the overall data density $P(\hat{\rho})$ of Tasks T2, T3 and T4 dataset images is much less, as compared to Task T1. Thus, the images of these three tasks have to be scaled down by interpolation.

For Task T2, the original dataset had 2500 good solar panel images (i.e., 50% of the overall dataset), which could bias the training and testing process. So, only 500 of them were included and the final dataset has 2700 images for training and 300 images for testing.

For Task T3, all the images were downsampled to 512×512 images and further divided into four 256×256 patches. The final dataset had around 7100 images for training and 500 images for testing. Lastly, the Task T4 images were also interpolated to size 256×256, followed by some data augmentation steps like image flipping and 90° rotation. The final dataset had around 1800 images for training and 200 images for testing. We applied some common data processing steps like noise addition, Gaussian blur addition and brightness changes. The entire data preprocessing step is shown in Fig. 2.

C. Network Architecture

The network architecture of SA U-Net++ is shown in Fig. 5. The SA U-Net++ architecture employs an encoder-decoder design. The model features four levels of encoder-decoder blocks, enumerated from top to bottom. Each level progressively captures finer details and nuances from the input image compared to its predecessor. Following these levels is the bottleneck layer, situated at the lowest level (5th level). In SA U-Net++, the number of convolutional filters increases from 32 filters at the initial/topmost level to 512 filters at the bottommost level. Despite its streamlined architecture with fewer convolutional layers and filters, SA U-Net++ outperforms numerous state-of-the-art (SOTA) models in terms of performance. A detailed insight into the architecture is given below. The encoder at each level employs a convolutional layer, succeeded by a maxpool layer and a dropout layer. The convolutional layer exploits local information from the input feature map to extract more intricate details useful to the task at hand with its filters. Each filter is trained to extract a specific feature from the input map, that integrates with the features from other filters to perform the segmentation. The maxpool layer downsamples the output feature map from the convolutional layer, preserving the most significant aspects of the map. The dropout layer addresses the issue of overfitting in the deeper levels of the models, which can arise due to an exponential surge in parametricity. It statistically disregards a certain percentage of the incoming feature parameters from its input layer based on the dropout criterion. This layer is implemented in the 3rd and 4th levels. At each level, the encoder's role is to discern and extract a more contextually significant, inherent, and condensed form of the data from the input, while discarding any extraneous information in the process. This can be mathematically represented as:

$$enc_i = \max\left(Conv_{3\times3}(f_i) * d_i\right) \tag{4}$$

In the above representation, f_i represents the input feature map at level i, acted upon by the convolutional layer $Conv_{3\times3}$ and the dropout percentage d_i, finally followed by the maxpool layer ($\max$), whose output is stored in enc_i.

The output of the last encoder (in 4th level) goes to the bottleneck layer at the lowest level. It consists of a single convolutional layer, which serves as the data sieve, refining the encoder output and passing only the necessary deep semantic representative data, which is devoid of any positional or orientational knowledge. The output from the bottleneck layer is directed to the decoder. It comprises of a 'combinational block', followed by a dropout layer (for 3rd and 4th levels), the self-attention block and the upsampling layer (for upsampling the feature maps level-wise to the original size). The combinational block and the self-attention blocks are discussed later in this section. The decoder performs additional feature extraction, followed by expansion to the original size in a level-wise manner, on the feature maps obtained from the previous level (rich in low-level intrinsic data) in combination with the skip connection output (rich in high-level global positional and orientational information), which is further enhanced by the self-attention module. The decoder's role is to combine deep semantic representation with positional embeddings for more accurate and meaningful segmentation. All this is indicated by the following equation:

$$dec_i = SA_{module}\left(f_i * d_i\right) \tag{5}$$

In the above equation, f_i represents the input feature map at level i, which is multiplied with the dropout coefficient d_i, followed by the self-attention module (SA_{module}). The result is stored in dec_i.

The dropout values for both encoder and decoders are: (i representing the level)

$$d_i = \begin{cases} 1, & \forall\ i < 3 \\ 0.8, & \forall\ i >= 3 \end{cases} \tag{6}$$

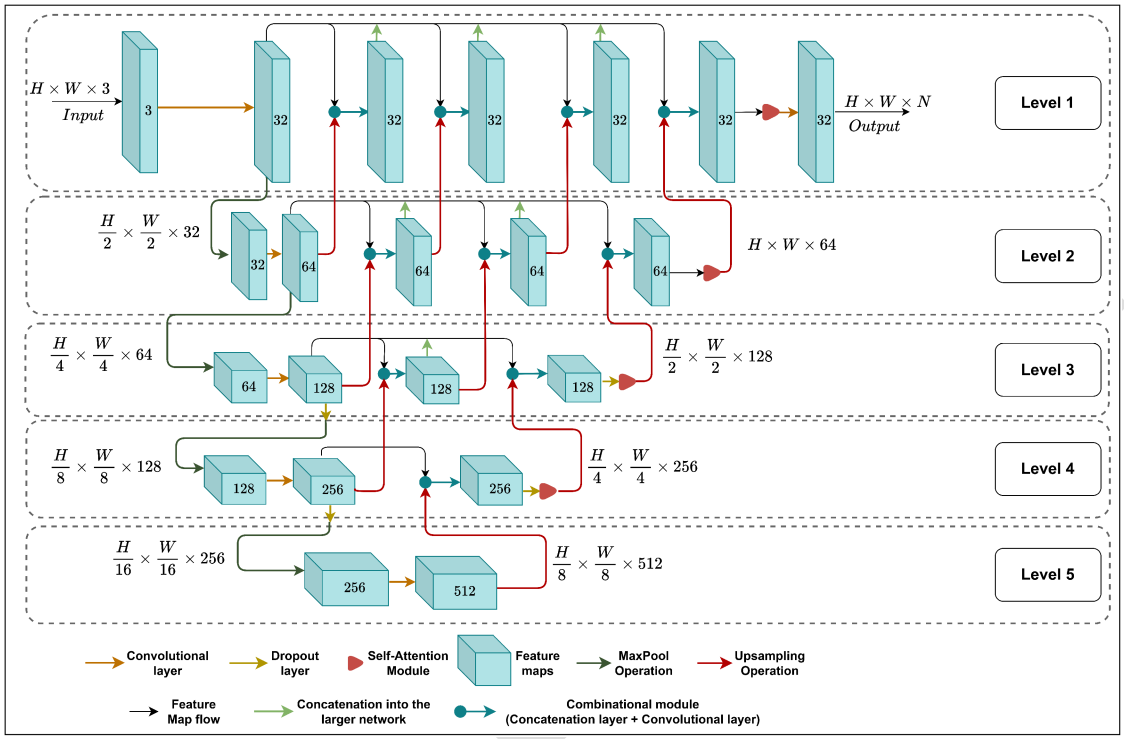

Fig. 5. The architecture of SA U-Net++. The dotted lines show the level-wise separations between the model structures. The arrows represent the different operations performed on the feature maps and along each level, the height and width dimensions of the feature maps do not change, until it reaches the end of the level, where it goes to the Upsampling layer, where these dimensions increase and marks as the transition between these layers. The number on each block represents the number of feature maps used. The convolutional layers use filters of size 3x3 by default. The output layer used a pointwise convolutional operation, with 'N' as the number of output classes.

The skip connections between the encoder and decoder blocks implement multiple combinational blocks for a gradual combination of the higher-level encoder data with its deep intrinsic representation in a level-wise manner for a better interpretation of the global scenario at the end of each level, thus providing a efficient segmentation. The outputs from these blocks are passed on to the subsequent block in the current level as well as is upsampled and passed on as an input to a block in the upper level.

Finally, the model inputs an image of shape 256 × 256 × 3 (RGB image) and the final arg-maxed output is of shape 256 × 256. Most convolutional layers used a 3 × 3 filter, except for the ones in the self-attention block and the output convolutional layer of the model, which used pointwise convolution. The loss function used is:

$$J = -\sum_{i=1}^{n} y_i \cdot \log(\hat{y}_i) \tag{7}$$

Where, J is the loss function, n is the number of training samples, y_i is the ground truth, and $\hat{y}_i$ is the prediction made by the model.

1) Combinational Block: The combinational block utilizes a concatenation layer, followed by a convolutional layer. This block is used in the skip pathway to merge outputs from the previous layers at the same level and from the layers at a lower level. Its purpose is to blend the deep semantic data extracted from lower levels with the high-level data, thereby encapsulating a more refined representation of global, positional, and orientational data. This simple, yet effective step-wise combination enhances the final output. It can be represented mathematically as:

$$CB_i^j = Conv_{3\times3}\left(x_i^j \ \phi \ U(u_{i-1}^j)\right) \tag{8}$$

In the above equation, U represents the upsampling operation, ϕ represents the concatenation operation and $Conv_{3\times3}$ represents the convolutional operation. CB_i^j represents the output of the jth combinational layer on the ith level, with x as the input from the same level and u as the input from a lower level after upsampling.

2) Self-Attention Block: The architecture of the self-attention (SA) block is depicted in Fig. 6. The input feature map to this module is split into three parts (vectors): Query (Q), Key (K) and Value (V) vectors respectively, all having the same dimension. The Query and Key vectors are passed through a pointwise convolution, followed by element-wise addition. This combination takes into account the global positioning of the pixels, and their relative importance in the feature map for the task at hand and accordingly assigns a value for every pixel. The following equation illustrates the operation:

$$a(x) = Conv_{1\times1}(K(x)) \otimes Conv_{1\times1}(Q(x)) \tag{9}$$

In (1), $Conv_{1\times1}$ represents the pointwise convolution operation, $\otimes$ denotes element-wise addition and $a(x)$ is its output,

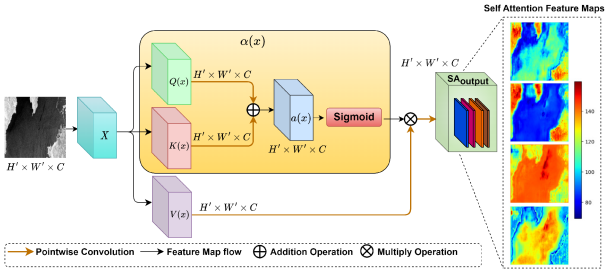

Fig. 6. The structure of the self-attention module. It displays the input image and its corresponding attention map representation in the output.

depicted as a function of input x. This operation is followed by a sigmoid activation function (σ) for normalising the output, which now acts as the attention weights ($\sigma(x)$) for the Value vector ((2) below).

$$\alpha(x) = \sigma(a(x)) = \frac{1}{1 + e^{-a(x)}} \tag{10}$$

Finally, the attention weights and the vector V are multiplied element-wise, and this output is passed through another pointwise convolution, giving the final output from the SA block ($SA_{output}(x)$). This operation can be summarised as follows:

$$SA_{output}(x) = Conv_{1\times1}\left(Conv_{1\times1}(V(x)) \otimes \alpha(x)\right) \tag{11}$$

The Self-Attention module differs from the cross-attention module in three main aspects. Firstly, the same input feature map is used in the Q and K vectors in the SA module. It can help in better identification of the essential pixels and also helps in establishing a crucial relationship among the pixels. In cross-attention, Q and K use two different maps. Then, we used sigmoid activation, instead of the more commonly used softmax function. Finally, they differ in the position of the attention modules placement. In the previous proposals, the cross-attention module had been used before the decoder, whereas in our model, the SA module is used after the decoder, before upsampling. This positioning can be justified by the fact that an SA module operates on a feature map based on a single input, thus this is an enhancement module based on self-interaction between the key-query pair and outputs the appropriate weightage for each object pixel in the feature map. The output from this module would be best when that single input has both deep semantic data and higher-dimension positional data embedded in it, and the SA module's current positioning at the end of each level would best achieve that.

D. Implementation Details

The implementation of the proposed model was carried out on the TensorFlow platform. We used Adam's Optimizer for optimization. To enhance the training process, we incorporated callbacks such as Reduce LR on Plateau and Early Stopping. The learning rates were set at 0.001, 0.0002, 0.00004, and 0.00001. The batch size used was 16, and the number of training epochs ranged from 30 to 70. The system configuration was 12 GB of RAM and a 15 GB NVIDIA Tesla T4 GPU.

IV. Result and Discussion

This section presents the performance evaluation of our proposed model for all the segmentation tasks. It gives a detailed analysis of our model in comparison with the following SOTA models: U-Net, U-Net++, U-Net 3+, Attention U-Net, R2 U-Net, RFBS Net, SegNet, LinkNet and DeepLabv3. The evaluation metrics used for this purpose are Precision (P), Recall (R), Jaccard Similarity (J), Dice Score (D) and Accuracy (A). The mean of these metrics over all the classes is used for multi-class segmentation.

A. Quantitative Analysis

Table I presents the performance of the models for each segmentation task based on five metrics. It clearly illustrates that the best performance for each task is returned by our proposed model SA U-Net++. For solar panel segmentation (Task T1), all models have a 93% dice score and only our model exceeded it and returned a 94% dice score. Whereas for defect segmentation (Task T2), the jaccard similarity and dice score results have risen to as high as 95% and 96% respectively. For the other two tasks as well, SA U-Net++ has been able to achieve a higher performance. The second best-performing model is U-Net 3+, with a 92% dice score for solar panel segmentation, 96% for defect segmentation, 84% for flood segmentation and 73% for oil spill segmentation. It returned better performance than both U-Net and U-Net++ models, as both these models don't use the full-scale input images, or high-level encoders (containing the spatial information). The complex skip interconnection among all the encoder and decoder modules in the U-Net 3+ model has been quite successful in the segmentation task, however, it made the model highly convoluted in terms of parametricity due to the utilization of numerous maxpool and upsampling operations, thus resulting in an increased inference time, as was observed during experimentations. These can give rise to scalability and implementation issues in real-time scenarios. Whereas, our proposed model with the SA module has been successful in improving upon the skip connection networks in U-Net++, keeping both parametricity and inference time comparatively low and returning a better performance than U-Net 3+. SA U-Net++ model performance on all of the tasks clearly shows its generalisation ability.

The Attention U-Net model also returned better results than U-Net. However, both these models seem to be victims of the semantic differences among various levels of the encoder-decoder blocks in the model. The worst performance is given by the SegNet and DeepLab models. While other models achieved a 92% dice score for solar panel segmentation or 94% dice score for defect segmentation, these two models were only able to achieve scores like 61% and 71%, respectively, for both these tasks. This can be attributed to the low parametricity and the lack of skip connections in SegNet (thus no transfer of global positioning data from encoder to decoder) and the use of AC in DeepLab. The skip connections play a major role in filling the semantic gap between the encoder and the decoders, and models like Attention U-Net, U-Net++ and U-Net 3+ have tried to

TABLE I
PERFORMANCE OF VARIOUS MODELS ON ALL THE TASKS

		SA U-Net++	U-Net++ [11]	Attention U-Net [13]	U-Net 3+ [12]	U-Net [10]	R2 U-Net [14]	RFBS Net [18]	SegNet [17]	LinkNet [16]	DeepLabv3 [15]
Task T1	*P*	0.95 ± 0.10	0.95 ± 0.10	0.95 ± 0.10	**0.96 ± 0.09**	0.95 ± 0.09	0.95 ± 0.10	0.95 ± 0.09	0.52 ± 0.34	0.95 ± 0.10	0.51 ± 0.34
	R	**0.94 ± 0.14**	0.93 ± 0.15	0.93 ± 0.15	0.92 ± 0.18	0.91 ± 0.19	0.93 ± 0.16	**0.94 ± 0.14**	**0.94 ± 0.24**	0.91 ± 0.20	0.93 ± 0.25
	J	**0.90 ± 0.16**	0.89 ± 0.17	0.89 ± 0.16	0.88 ± 0.19	0.87 ± 0.20	0.89 ± 0.17	0.89 ± 0.17	0.52 ± 0.34	0.87 ± 0.21	0.51 ± 0.34
	D	**0.94 ± 0.14**	0.93 ± 0.14	0.93 ± 0.14	0.92 ± 0.17	0.91 ± 0.18	0.93 ± 0.15	0.93 ± 0.15	0.61 ± 0.33	0.91 ± 0.19	0.60 ± 0.33
	A	**0.96 ± 0.06**	**0.96 ± 0.07**	**0.96 ± 0.06**	**0.96 ± 0.08**	0.95 ± 0.08	**0.96 ± 0.07**	0.94 ± 0.07	0.52 ± 0.34	0.95 ± 0.09	0.51 ± 0.34
Task T2	*P*	0.97 ± 0.07	0.95 ± 0.10	0.95 ± 0.11	0.97 ± 0.07	0.95 ± 0.10	0.95 ± 0.10	0.94 ± 0.07	**1.00 ± 0.00**	0.95 ± 0.10	0.94 ± 0.12
	R	**0.98 ± 0.03**	0.96 ± 0.04	**0.98 ± 0.03**	**0.98 ± 0.03**	0.97 ± 0.04	**0.98 ± 0.02**	0.94 ± 0.05	0.71 ± 0.11	0.95 ± 0.05	0.77 ± 0.12
	J	**0.95 ± 0.09**	0.92 ± 0.12	0.93 ± 0.12	0.94 ± 0.08	0.92 ± 0.12	0.94 ± 0.11	0.90 ± 0.08	0.71 ± 0.11	0.91 ± 0.11	0.71 ± 0.20
	D	**0.96 ± 0.08**	0.94 ± 0.11	0.95 ± 0.11	**0.96 ± 0.08**	0.94 ± 0.11	0.95 ± 0.11	0.94 ± 0.08	0.71 ± 0.11	0.93 ± 0.11	0.73 ± 0.21
	A	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Task T3	*P*	**0.89 ± 0.12**	0.87 ± 0.12	0.86 ± 0.14	0.88 ± 0.13	0.83 ± 0.14	0.86 ± 0.14	0.86 ± 0.13	0.88 ± 0.13	0.87 ± 0.13	0.76 ± 0.18
	R	**0.89 ± 0.11**	0.87 ± 0.12	0.87 ± 0.13	**0.89 ± 0.12**	0.83 ± 0.13	0.87 ± 0.13	0.87 ± 0.12	0.88 ± 0.13	0.88 ± 0.13	0.77 ± 0.17
	J	**0.81 ± 0.20**	0.77 ± 0.21	0.76 ± 0.22	0.79 ± 0.22	0.70 ± 0.23	0.76 ± 0.24	0.76 ± 0.22	0.78 ± 0.23	0.78 ± 0.23	0.59 ± 0.29
	D	**0.86 ± 0.18**	0.83 ± 0.18	0.82 ± 0.20	0.84 ± 0.20	0.77 ± 0.21	0.82 ± 0.21	0.82 ± 0.19	0.84 ± 0.20	0.83 ± 0.20	0.66 ± 0.28
	A	**0.87 ± 0.16**	0.85 ± 0.17	0.84 ± 0.18	0.86 ± 0.18	0.79 ± 0.19	0.84 ± 0.19	0.84 ± 0.18	0.85 ± 0.18	0.85 ± 0.19	0.69 ± 0.27
Task T4	*P*	0.83 ± 0.14	**0.85 ± 0.12**	0.84 ± 0.12	0.82 ± 0.15	0.84 ± 0.12	0.83 ± 0.11	0.82 ± 0.13	0.68 ± 0.15	0.75 ± 0.13	0.82 ± 0.12
	R	**0.83 ± 0.12**	0.79 ± 0.13	0.77 ± 0.14	**0.83 ± 0.12**	0.79 ± 0.15	0.79 ± 0.14	0.74 ± 0.15	0.71 ± 0.13	0.74 ± 0.13	0.69 ± 0.14
	J	**0.70 ± 0.18**	0.68 ± 0.18	0.65 ± 0.18	**0.70 ± 0.18**	0.67 ± 0.19	0.66 ± 0.18	0.60 ± 0.17	0.48 ± 0.15	0.56 ± 0.16	0.63 ± 0.16
	D	**0.74 ± 0.18**	0.71 ± 0.17	0.68 ± 0.18	0.73 ± 0.18	0.71 ± 0.18	0.69 ± 0.18	0.63 ± 0.17	0.51 ± 0.15	0.59 ± 0.16	0.65 ± 0.16
	A	**0.96 ± 0.08**	0.95 ± 0.08	0.95 ± 0.08	0.95 ± 0.07	0.95 ± 0.08	0.95 ± 0.08	0.93 ± 0.09	0.93 ± 0.09	0.94 ± 0.09	0.94 ± 0.09

*Individual cell values are in the format of Mean ± Standard Deviation. The bold values represent the best-performing model on a particular metric.

improve on this skip connectivity of the models, by introducing gating signals or redesigning the skip connections.

B. Qualitative Studies

After quantitative studies, the qualitative studies of our model are presented for all the tasks. There are two parts to this study. The first part includes two input images from each task, followed by their corresponding output by each model along with the ground truth segmentation. The second part discusses an erroneous prediction by SA U-Net++ for each task. This study will help understand the strengths and weaknesses of our proposed model.

The first part of this study is presented in Fig. 7 for all tasks. Fig. 7(a) of Task T1, the SOTA models had difficulty in separating the two solar panel regions properly and for Fig. 7(b) of Task T1, the SOTA models returned many false positives. They predicted the surrounding objects as the solar panels. Task T2 figures show the zoomed-in view of the model's defect segmentation output, which otherwise is not properly visible. Fig. 7(a) of Task T2 shows that the SOTA models misclassified some of the pixels in the broken panel, and classified them as missing cells. Task T3 figures show some major prediction casualties, with many false positives and false negatives per class, for instance, in Fig. 7(a), the models failed to properly segment out the respective class clusters, leading to some significant leaks (i.e., misclassifications) into other classes. Finally, for Task T4, the SOTA models again suffered from misclassifications and produced outputs with some border inaccuracies for both the input images. Thus, all the figures show that SA U-Net++ gave the closest outputs to each task, when compared to other models, and thus is the best-performing model.

The second part of this study is shown in Fig. 8. The main reason for the prediction error in Task T1 stemmed from the pronounced texture similarity between the object positioned on the left side of the image and a solar panel; however, the solar panel on the right side is well segmented. In the case of Task T2, the error is not in the segmentation mask, but in the type to which it is assigned; instead of assigning to the broken cell class, it is assigned to the missing cell class, due to the similarity in their appearance. For Task T3, the error lies in the lack of precision with which the border of the image objects is segmented. Lastly, the error observed in Task T4 primarily stems from the semi-segmented oil spills, attributed to variations in intensity levels within the oil spill region, which happens to be a rare occurrence. When the entire Qualitative Analysis is considered, it can be concluded that our SA U-Net++ model effectively segments aerial and satellite images regardless of the task or the size of the target object it is trained on (for instance, Task T2 involves segmenting tiny defects in solar panels).

TABLE II
ABLATION STUDY RESULTS ON ATTENTION MODULE

Attention Level	Precision	Recall	Jaccard Similarity	Dice Score
Level-1	0.96 ± 0.09	0.97 ± 0.03	0.93 ± 0.11	0.95 ± 0.10
Level-1, 2	0.96 ± 0.08	0.97 ± 0.04	0.94 ± 0.10	**0.96 ± 0.09**
Level-1, 2, 3	0.95 ± 0.09	0.97 ± 0.04	0.92 ± 0.11	0.94 ± 0.10
Level-1, 2, 3, 4 (proposed)	**0.97 ± 0.07**	**0.98 ± 0.03**	**0.95 ± 0.09**	**0.96 ± 0.08**
Level-1, 2, 3, 4, 5	0.93 ± 0.13	0.97 ± 0.04	0.91 ± 0.14	0.93 ± 0.13
CBAM attention	0.96 ± 0.06	0.96 ± 0.04	0.93 ± 0.08	0.95 ± 0.07
Squeeze and Excite attention	0.96 ± 0.09	**0.98 ± 0.03**	0.94 ± 0.10	0.95 ± 0.09

*Individual cell values are in the format of Mean ± Standard Deviation

C. Ablation Studies

In this section, we presented an ablation study on our proposed model. The ablation study aims to assess the capabilities of different variants of SA U-Net++ and validate the performance of the proposed model. There are two parts to this study, in the first part, we performed the ablation study on attention (Table II) and in the second part, it is performed on the dropout layer (Table III). The ablation study is performed on solar panel defect segmentation.

1) Ablation Study on Attention: The ablation study on the attention modules investigates seven different variants of SA U-Net++, each employing a distinct type and placement of

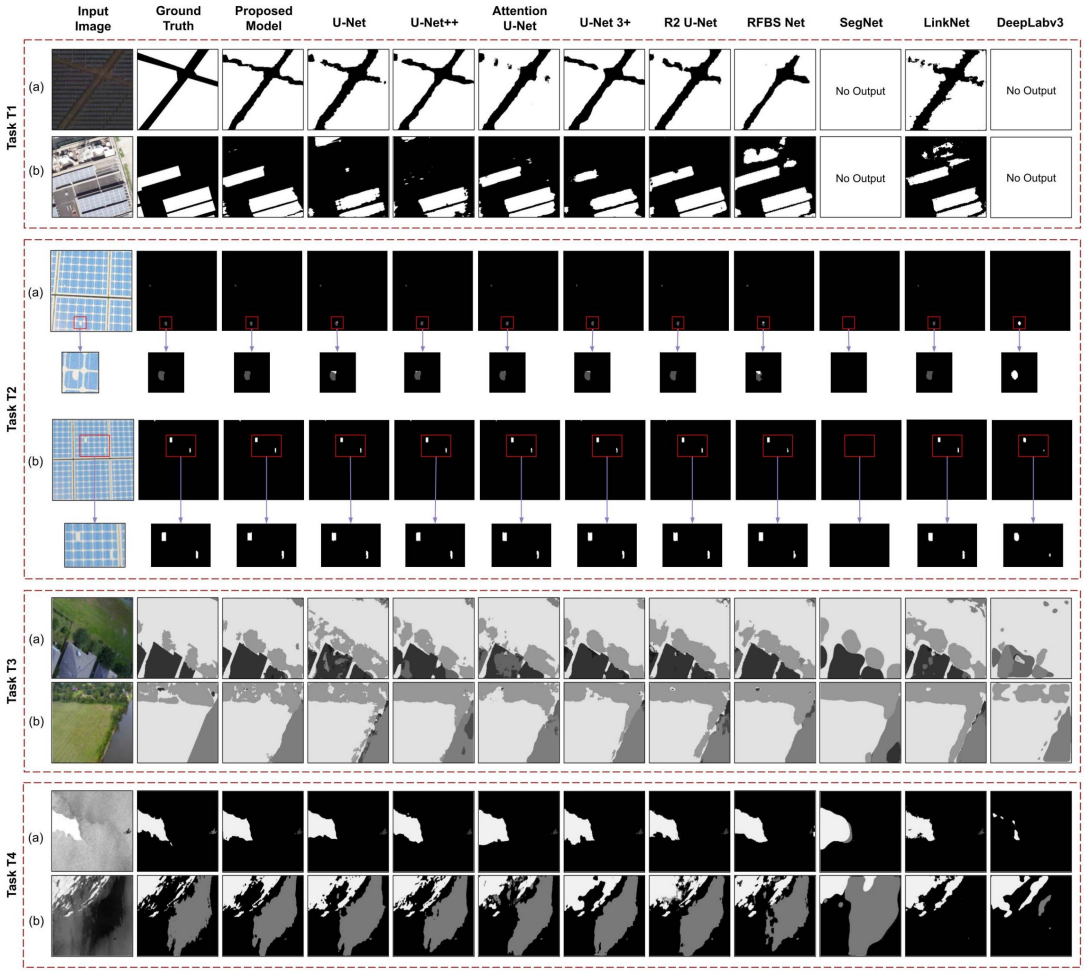

Fig. 7. Qualitative analysis of SA U-Net++ on all the tasks in comparison to other models. Two images per task are included for this analysis and the output of all the models, along with the ground truth segmentation, is displayed.

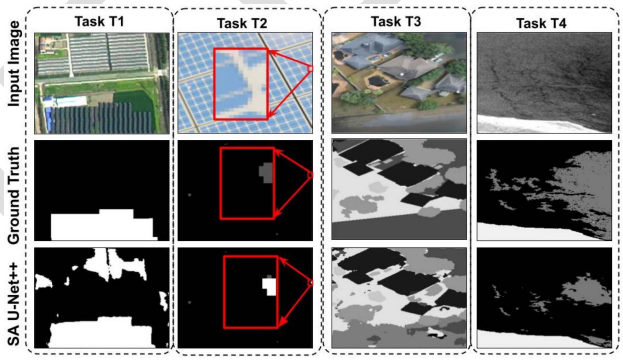

Fig. 8. Qualitative analysis showing the error in the prediction made by SA U-Net++ on all the tasks.

attention network. The initial five variants incorporate the SA module as follows: the first variant utilizes the SA module solely in level 1, the second variant incorporates it in levels 1 and 2, and so forth, until the fifth variant employs it across all five levels. The last two variants use two different types of specialised networks in place of the SA module: CBAM (Convolutional Block Attention Module) [33] and SE (Squeeze-and-Excite Networks) [34]. CBAM is a module that makes use of both channel and spatial attention serially; whereas, the SE network uses channel attention with the help of a global pooling operation. The performance results are shown in Table II showing the standard deviation around the mean.

TABLE III
ABLATION STUDY RESULTS ON DROPOUT LAYER

	Precision	Recall	Jaccard Similarity	Dice Score
No dropout	0.94 ± 0.11	0.96 ± 0.04	0.91 ± 0.12	0.94 ± 0.12
Level-4	0.96 ± 0.09	0.97 ± 0.03	0.94 ± 0.10	0.95 ± 0.09
Level-4, 3 (proposed)	**0.97 ± 0.07**	**0.98 ± 0.03**	**0.95 ± 0.09**	**0.96 ± 0.08**
Level-4, 3, 2	0.95 ± 0.1	0.97 ± 0.04	0.93 ± 0.12	0.95 ± 0.11
Level-4, 3, 2, 1	0.96 ± 0.1	0.97 ± 0.04	0.93 ± 0.12	0.95 ± 0.11

*Individual cell values are in the format of Mean ± Standard Deviation

Table II shows that the precision in all of the variants is roughly the same, and the recall in the last three variants is lower than in the first four, denoting a higher false negative rate. However, the main insight is provided by the dice score.

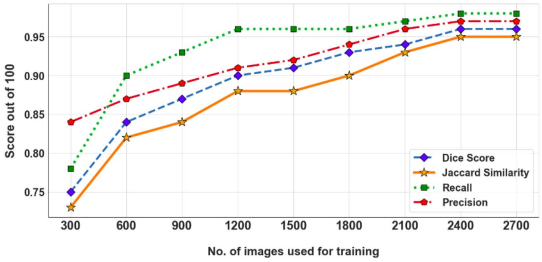

Fig. 9. Plot showing the performance of SA U-Net++ trained with datasets of different sizes.

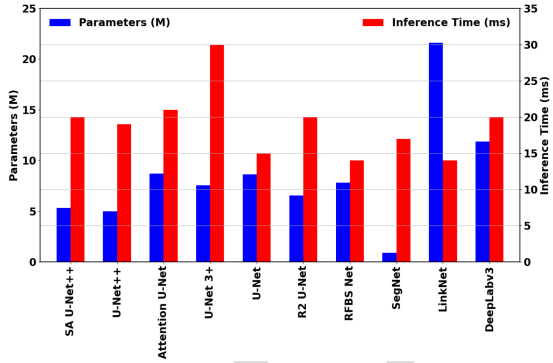

Fig. 10. Plot showing the no. of parameters and the inference time comparison of all the models under consideration.

It can be seen that CBAM and SE networks returned a lower performance than the ones using the SA modules. And the best variant of SA U-Net++ is the one using four SA modules (the proposed variant). As can be expected, increasing the number of SA modules and incorporating them at various levels will result in a gradual increase in attention towards the necessary pixel of the feature maps. However, using the module in the bottleneck layer does not turn out to be advantageous, as the raw, low-level encoded output hardly has any high-level positional embedding in it for the self-attention mechanism to give weightage to.

2) Ablation Study on Dropout: Dropout is used in deeper layers to avoid overfitting. Using a larger number of dropout layers results in convergence problems during training, whereas a lower number of dropouts can result in overfitting. Thus this ablation study will help us determine the optimal number of dropout layers for the model. However, dropouts are not applied to the bottleneck layer (i.e., 5th level), as it contains the raw and finest intrinsic representation of the input data, which will be spoilt if dropped out. Five variants of SA U-Net++ were tested in this study: the first variant uses no dropout layer, the second one uses it only in the 4th level, the third one uses it in the 3 rd and 4th levels, and so on till the last one uses it in all the levels (except for the bottleneck layer). The performance results are shown in Table III.

Table III indicates no noticeable observations in the precision and recall results. The dice score shows that using dropout layers in the 3 rd and 4th levels (the 3 rd model) gives the best results.

D. Minimum Data-Size Identification Experiment

After the ablation studies, we present an experiment, where our proposed model is trained on datasets of different sizes and their test results are accordingly reported. This study is performed on solar panel defect segmentation and is intended to check two important criteria: firstly, determining whether the model is prone to overfitting, and secondly, identifying the minimum required size of the training dataset to achieve satisfactory results. The latter parameter is contingent upon the complexity of the task at hand. A larger training data corresponds to a better training process. So, using a smaller training dataset will degrade the performance, however, the extent of degradation is to be noted. Drastic degradation implies that the model overfitted the training data to some degree. Fig. 9 shows the performances of nine SA U-Net++ models trained on training datasets of different sizes: the first model trained on 300 images, the second model on 600 images, and so on till the ninth model trained on 2700 images. As can be seen, the performance drop between the first and the last model is even less than 20% and the maximum performance drop between two consecutive models in the plot is hardly 4%, which shows the chances of overfitting in our model is less. This drop is expected due to the use of fewer data for training. Also, it can be concluded for solar panel defect segmentation that there should be at least 2400 images in the training data (70% of the original dataset) for getting a satisfactory performance. More images should be included for further performance enhancement.

E. Parametricity and Inference Time Test

Finally, we have shown a comparison of the number of parameters and the inference time required for each model in Fig. 10. Inference time is the time required by the model to provide the segmentation output for one input image and is calculated as the difference between the system time at which the model receives the input image and the system time at which the model returns the output. Fig. 10 clearly shows that the U-Net 3+ model has the highest average inference runtime of 30 msec. This is attributed to the numerous maxpool and upsampling operations performed by the model at each level, which do not contribute to the parameter count, thus keeping it comparatively low, but significantly increasing the execution time. Models like RFBS Net, U-Net and LinkNet have a lower inference time, despite having a higher number of parameters, because of the absence of multi-level skip connection or gating signals like in U-Net++, U-Net 3+ or attention U-Net. However, LinkNet has the highest parameter count because of the use of a large number of convolutional filters. Finally, it is seen that the number of parameters required by U-Net++ in our setting is approximately half of that used in [11]. When both these criteria (number of parameters and inference time) are taken into account, along with these models' performance, it can undoubtedly be concluded that our proposed model SA U-Net++ performs the best, successfully maintaining a trade-off between the performance, the number of parameters and the inference time.

In future, we propose to work on and improve self-attention based methods and architectures to enhance efficiency and provide a performance boost. In addition, we plan to extend the applicability of our methods to other environmental issues such as wildfire detection, cropland flooding, hurricane damage, etc. Self-attention has huge potential, which needs to be explored and we plan to extend the applicability of self-attention to many other computer vision domains.

V. Conclusion

In this paper, we dealt with the task of environmental monitoring and protection and took the help of DL methods for solving four important tasks in this domain: solar panel segmentation, solar panel defect identification, flood segmentation and oil spill segmentation. Accordingly we have proposed SA U-Net++, a modified variant of the U-Net++ segmentation model with redesigned skip connections and equipped with a self-attention mechanism. This architecture boosted the performance of U-Net++, which surpassed the SOTA models. The model is generalisable, lightweight and efficient, thus being scalable and at the same time having low time and space complexity. SA U-Net++ has given the best results among all the models, for all the tasks, and its performance indicates that self-attention has huge potential, which can be explored further in future with different backbones/architectures and for many different use cases.

References

[1] S. Yadav et al., "Environmental education for sustainable development," in *Natural Resources Conservation and Advances for Sustainability*. Amsterdam, The Netherlands: Elsevier, 2022, pp. 415–431.

[2] P. Ndimele et al., "Remediation of crude oil spillage," in *The Political Ecology of Oil and Gas Activities in the Nigerian Aquatic Ecosystem*. Norwell, MA, USA: Academic Press, 2018, pp. 369–384.

[3] S. F. Singer, "Global effects of environmental pollution," *Eos Trans. Amer. Geophysical Union*, vol. 51, no. 5, pp. 476–478, 1970.

[4] F. Barbir, T. N. Veziroğlu, and H. J. Plass Jr, "Environmental damage due to fossil fuels use," *Int. J. Hydrogen Energy*, vol. 15, no. 10, pp. 739–749, 1990.

[5] S. Shafiee and E. Topal, "When will fossil fuel reserves be diminished?," *Energy Policy*, vol. 37, no. 1, pp. 181–189, 2009.

[6] F. S. Memon, "Catastrophic effects of floods on environment and health: Evidence from Pakistan," in *Memon, FS Sharjeel, MY*, 2015, pp. 72–84.

[7] J.-P. W. Desforges et al., "Immunotoxic effects of environmental pollutants in marine mammals," *Environ. Int.*, vol. 86, pp. 126–139, 2016.

[8] R. Li et al., "Deep learning based imaging data completion for improved brain disease diagnosis," in *Med. Image Comput. Computer-Assist. Interv. 2014, 17th Int. Conf.*, 2014, pp. 305–312.

[9] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 3431–3440.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Med. Image Comput. Computer-Assist. Interv. 2015, 18th Int. Conf.*, 2015, pp. 234–241.

[11] Z. Zhou et al., "UNet : A. nested U-Net architecture for medical image segmentation," in *Deep Learn. Med. Image Anal. Multimodal Learn. Clin. Decis. Support, 4th Int. Workshop, 2018, 8th Int. Workshop, ML-CDS 2018, Held in Conjunction 2018*, 2018, pp. 3–11.

[12] H. Huang et al., "UNet 3 : A full-scale connected unet for medical image segmentation," in *ICASSP 2020-2020 IEEE Int. Conf. Acoust., Speech Signal Process.*, 2020, pp. 1055–1059.

[13] O. Oktay et al., "Attention U-Net: Learning where to look for the pancreas," 2018, *arXiv:1804.03999*.

[14] M. Alom et al., "Recurrent residual U-net for medical image segmentation," *J. Med. Imag.*, vol. 6, no. 1, pp. 014006–014006, 2019.

[15] M. Liu et al., "Comparison of multi-source satellite images for classifying marsh vegetation using DeepLabV3 plus deep learning algorithm," *Ecological Indicators*, vol. 125, 2021, Art. no. 107562.

[16] P. Sameer, V. Anirudh Raj Kaushik, and R. Indrakanti, "Brain tumor segmentation using LinkNet," in *2022 2nd Int. Conf. Next Gener. Intell. Syst.*, 2022, pp. 1–5.

[17] T. Chen et al., "Pavement crack detection and recognition using the architecture of SegNet," *J. Ind. Inf. Integr.*, vol. 18, 2020, Art. no. 100144.

[18] R. Faghihpirayesh et al., "Deep Learning Framework for Real-Time Fetal Brain Segmentation in MRI," in *Perinatal, Preterm Paediatric Image Anal., 7th Int. Workshop 2022, Held in Conjunction with MICCAI 2022*, 2022, pp. 60–70.

[19] L. Zhuang, Z. Zhang, and L. Wang, "The automatic segmentation of residential solar panels based on satellite images: A cross learning driven U-net method," *Appl. Soft Comput.*, vol. 92, 2020, Art. no. 106283.

[20] R. Zhu et al., "Deep solar PV refiner: A detail-oriented deep learning network for refined segmentation of photovoltaic areas from satellite imagery," *Int. J. Appl. Earth Observation Geoinformation*, vol. 116, 2023, Art. no. 103134.

[21] M. A. Wani and T. Mujtaba, "Segmentation of satellite images of solar panels using fast deep learning model," *Int. J. Renewable Energy Res.*, vol. 11, no. 1, 2021, Art. no. 31–45.

[22] J. Camilo et al., "Application of a semantic segmentation convolutional neural network for accurate automatic detection and mapping of solar photovoltaic arrays in aerial imagery," 2018, *arXiv:1801.04018*.

[23] P. Parhar et al., "HyperionSolarNet: Solar panel detection from aerial images," 2022, *arXiv:2201.02107*.

[24] H. Chen et al., "Solar cell surface defect inspection based on multispectral convolutional neural network," *J. Intell. Manuf.*, vol. 31, pp. 453–468, 2020.

[25] M. Vlaminck et al., "Region-based CNN for anomaly detection in PV power plants using aerial imagery," *Sensors*, vol. 22, no. 3, 2022, Art. no. 1244.

[26] Y. S. Balc?o?lu, B. Sezen, and C. Cubukcu Cerasi, "Solar cell busbars surface defect detection based on deep convolutional neural network," *IEEE Latin America Trans.*, vol. 21, no. 2, pp. 242–250, Feb. 2023.

[27] D. Hernández et al., "Flood detection using real-time image segmentation from unmanned aerial vehicles on edge-computing platform," *Remote Sens.*, vol. 14, no. 1, 2022, Art. no. 223.

[28] M. Rahnemoonfar, T. Chowdhury, A. Sarkar, D. Varshney, M. Yari, and R. R. Murphy, "Floodnet: A high resolution aerial imagery dataset for post flood scene understanding," *IEEE Access*, vol. 9, 2021, pp. 89644–89654.

[29] S. T. Yekeen, A. Balogun, and K. B. Wan Yusof, "A novel deep learning instance segmentation model for automated marine oil spill detection," *ISPRS J. Photogrammetry Remote Sens.*, vol. 167, 2020, pp. 190–200.

[30] M. Krestenitis et al., "Oil spill identification from satellite images using deep neural networks," *Remote Sens.*, vol. 11, no. 15, 2019, Art. no. 1762.

[31] M. Krestenitis et al., "Early identification of oil spills in satellite images using deep CNNs," in *MultiMedia Modeling, 25th Int. Conf.*, 2019, pp. 424–435.

[32] H. Jiang et al., "Multi-resolution dataset for photovoltaic panel segmentation from satellite and aerial imagery," *Earth System Sci. Data*, vol. 13, no. 11, pp. 5389–5401, 2021.

[33] S. Woo et al., "CBAM: Convolutional block attention module," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 3–19.

[34] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7132–7141.

[35] T. Bao et al., "MIAD: A. maintenance inspection dataset for unsupervised anomaly detection," 2022, *arXiv:2211.13968*.

Tuhinangshu Gangopadhyay received the B.Tech. degree in computer science and engineering from the Government College of Engineering and Leather Technology, Kolkata, India. He is currently an Erasmus Mundus Joint master's student in medical imaging and applications with the University of Burgundy, Dijon, France, University of Cassino, Cassino, Italy, and University of Girona, Girona, Spain. He did his internship with Jio Institute, Navi Mumbai, India. His research interests include a broad spectrum of topics, such as AI and ML, computer vision, and biomedical image analysis.

Tanushree Meena received the master's and Ph.D. degrees from the Indian Institute of Technology (Banaras Hindu University), Varanasi, India. She is currently a Postdoc with Jio Institute, Navi Mumbai, India. She made significant contributions to the field of computer science and engineering, including publications in IEEE, Elsevier, and Springer. Her research interests mainly include artificial intelligence, computer vision, human computer interaction, medical imaging, and sensor data analysis.

Debojyoti Pal received the B.Tech. degree in computer science and engineering from the Government College of Engineering and Leather Technology, Kolkata, India. He was a RA with Jio Institute, Navi Mumbai, India. He is currently with Washington University in St. Louis, St. Louis, MO, USA.

Sudipta Roy is currently an Assistant Professor of artificial intelligence & data science with Jio Institute, Navi Mumbai, India. Throughout his career, he has made significant contributions to the field of computer science and engineering, with more than 80 publications in reputable international journals and conferences, including *Cell*, *EBioMedicine*- The Lancet (co-published), IEEE TRANSACTIONS ON MEDICAL IMAGING, MedIA, CMPB, CBM, JNM, EJNMMI, CVPR, ICCV, MICCAI, ISBI, PReMI, and ICPR.