III. Model documentation and write-up

Information included in this section may be shared publicly with challenge results. You can respond to these questions in an e-mail or as an attached file. Please number your responses.

1. Who are you (mini-bio) and what do you do professionally? If you are on a team, please complete this block for each member of the team.
   My name is Azin Al Kajbaf. I am a postdoctoral research fellow at Johns Hopkins University and the National Institute of Standards and Technology (NIST). As a postdoctoral fellow, I am leveraging data science techniques and geospatial analysis to collaborate in projects to support community resilience planning through the development of methods and tools that evaluate the economic impacts of disruptive events.

   My name is Kaveh Faraji. I am a Ph.D. candidate at the University of Maryland, and I work in the area of Disaster Resilience. The primary focus of my work is using geospatial analysis, statistical analysis, and machine learning in probabilistic risk assessment of critical infrastructure against natural hazards such as floods, sea level rise and hurricanes.

2. What motivated you to compete in this challenge?
   We implement machine learning and deep learning methods in our research projects. We are interested to learn more about these approaches and their real-world application. Competing in these challenges allows us to gain experience with different machine learning and deep learning techniques that we can potentially employ in our research projects as well.

3. High level summary of your approach: what did you do and why?
   Our best scored solution is an ensemble of 19 U-net convolutional neural network models. We used different U-net structures with different number of layers, and several loss functions such as dice loss, focal loss and combination of these two. We also used multiple strategies for partitioning of the data.

4. Do you have any useful charts, graphs, or visualizations from the process?
   No.

5. Copy and paste the 3 most impactful parts of your code and explain what each does and how it helped your model.
   We used a combination of different loss models (e.g., dice loss, focal loss, and squared dice loss) which was helpful in improving our score.

```
function loss = forwardLoss(layer, Y, T)

    % The predictions Y and the training targets T.
    smooth = 1;
    smooth2 = 1e-6;
    A= reshape(Y,1,[]);
    B = reshape(T,1,[]);
```

```
            bce = -mean(B.*log(A+smooth2) + (1-B).*log(1-A+smooth2));
            alpha = layer.Alpha;
            gamma = layer.Gamma ;
            sq = layer.Sq ;
            focal_loss = (alpha*(1-exp(-bce)).^gamma).*bce;

            intersection = sum(A.* B);

            if sq
            dice_loss = 1 - ((2 * intersection + smooth) / ((sum(A.^2.) +
sum(B.^2.)) +smooth));
            else
            dice_loss = 1 - ((2 * (intersection) + smooth) / ((sum(A) +
sum(B)) +smooth));
            end

            if layer.loss_type == 1
                loss =  dice_loss + focal_loss;
            elseif layer.loss_type == 2
                loss =  dice_loss;
            elseif layer.loss_type == 3
                loss = focal_loss;
            end

        end
```

Another important part of the code was using different U-net structures and using different partitioning strategies for the training data.

6. Please provide the machine specs and time you used to run your model.
   - CPU (model): AMD Ryzen9 7950X Processor
   - GPU (model or N/A): RTX 4090
   - Memory (GB): 64 GB
   - OS: Linux
   - Train duration: ~ 144 hours
   - Inference duration: ~ 1 hour

7. Anything we should watch out for or be aware of in using your model (e.g. code quirks, memory requirements, numerical stability issues, etc.)?
   No.

8. Did you use any tools for data preparation or exploratory data analysis that aren't listed in your code submission?
   No.

9. How did you evaluate performance of the model other than the provided metric, if at all?
   We did not use any other metric.
10. What are some other things you tried that didn't necessarily make it into the final workflow (quick overview)?

We started working on an XGBoost model, but due to time limitation, it did not end up in the final ensemble.

11. If you were to continue working on this problem for the next year, what methods or techniques might you try in order to build on your work so far? Are there other fields or features you felt would have been very helpful to have?
    If we were to continue working on this problem, we would have tried an ensemble of boosted trees and convolutional neural networks.

12. What simplifications could be made to run your solution faster without sacrificing significant accuracy?
    One thing that can be tested is using a higher learning rate and a smaller number of epochs (not guaranteed to not cause divergent behavior in loss function).