Flood Map Prediction Challenge, phase two

E.J Team

March 30, 2024

1 Performance Evaluation

1.1 Model performance: AUC ROC

Explain how you optimize the performance of your model. What methods did you use to process your data, select your features, choose your model and hyperparameters?

We first found that two major difficulties of the challenge were (i) the size of images in the data (up to 5953x3936 pixels per image), and (ii) the strong imbalance between classes (most labels are zeros, and only a few are ones). To address both of these issues, we split images into smaller patches. These patches are much smaller than the original image and were easier to pass as input to our neural networks, while allowing for a larger batch size and therefore more stable training. Another benefit of the patch approach is the fact that we can choose to keep only certain patches for training: in practice, we kept only the patches that had floods at some timestep within the training period or were close to one of such patches. Moreover, in order to have the same resolution for all inputs in our training data, we upsampled the low-resolution spatio-temporal features to match the resolution of geospatial features. We used an interpolation technique based on implicit neural representations [Naour et al., 2023].

We chose to use convolutional neural networks (CNNs) as they are a standard approach for deep learning-based image classification. We also tried vision transformers (ViTs), but they didn't reach the same performance as CNNs, which is likely due to the type of information they encode: ViTs are better at capturing global information while CNNs are very good at pattern recognition [Raghu et al., 2021]. We tried different convolutional architectures, including a UNet [Ronneberger et al., 2015], but in the end chose to keep a simpler and shallower CNN architecture, as it was able to reach similar performance when carefully tuned but was more frugal.

Regarding hyperparameter tuning, we adjusted hyperparameters manually while following an incremental strategy as proposed by Godbole et al. [2023]. We first tuned our model architecture (the size of the kernel and the number of convolutional layers). As both kernel size and network depth can interact with the learning rate, we tuned each parameter over multiple learning rates in order to fairly compare them, i.e., we treat the learning rate as a nuisance parameter [Godbole et al., 2023].

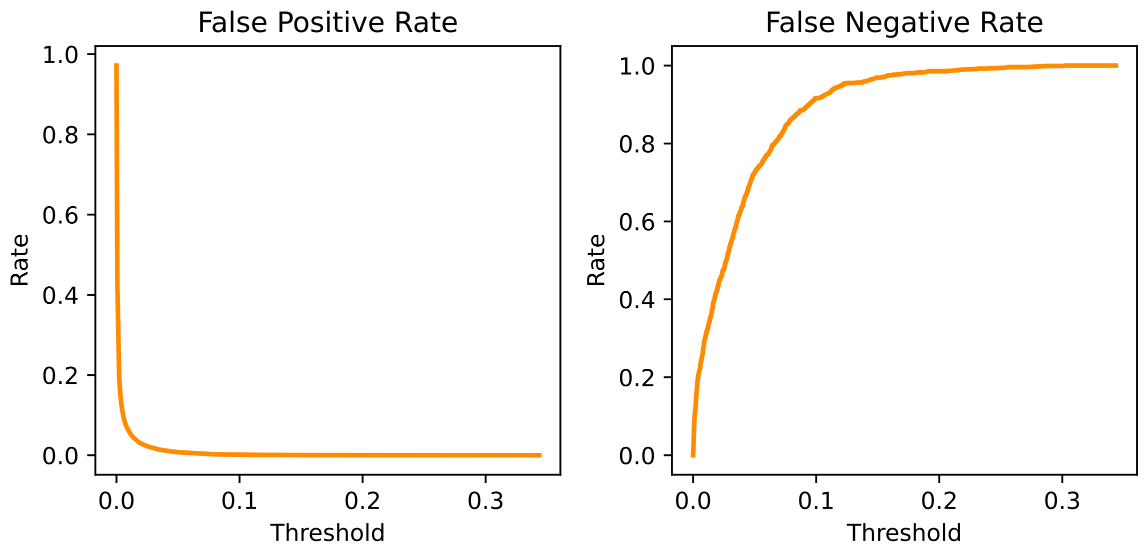


Figure 1: Error rates as a function of classification threshold.

1.2 Use case performance: False positive rate

False positive: $y_{pred} \geq 0.9$ when $y = 0$.

According to this definition, we do not get any false positives, as none of our predicted probabilities surpass 0.9. This is not surprising as there are not many flooded areas in the training data (both spatially and temporally) and the model cannot usually be certain that a flood event will happen. Yet, we can adjust the classification threshold and observe its effect on the false positives and false negatives. In Figure 1, we show the false positive and false negative rates as a function of the threshold. We can see that both rates have most of their variation at low threshold values. Moreover, the false negative rate increases more slowly that the false positive rate decreases, indicating that the model gets very averse to false positives before it starts being overly conservative and predict mostly false negatives.

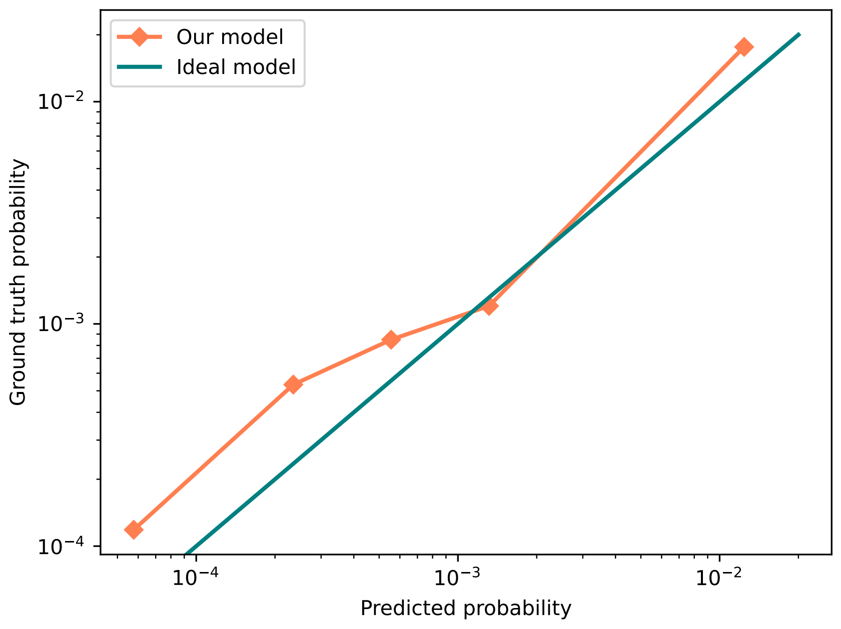


Figure 2: Calibration plot of our model.

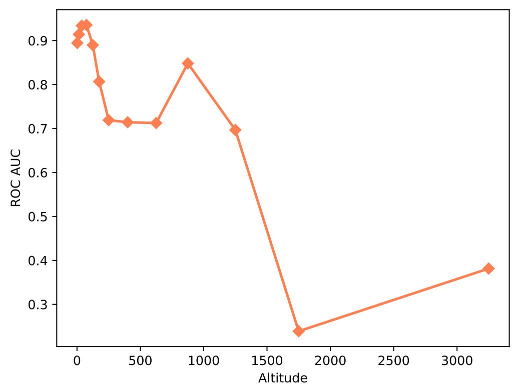
2 Robustness

2.1 Calibration

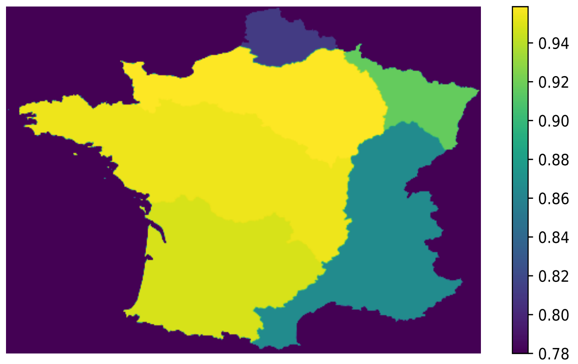
Explain how you calibrated your model.

We noted that it is hard to get a stable training and end up with calibrated probabilities from whole training images, as the label distributions are highly imbalanced. To get better calibrated probabilities, we split the training data into patches that had at least one occurrence of flooding in the training period (as well as patches close to those) and those that did not and were therefore very unlikely to see any flood event in the future, most likely because they were far from any major river or body of water. We passed the former as input to our convolutional network trained with the binary cross-entropy loss function which is known to give well-calibrated probabilities, at least when classes are not too imbalanced, as it directly performs maximum likelihood estimation. For the rest of the patches, we assigned a prediction of 0 to them as it was very likely that almost all would not be subject to flood in the future.

We report in Figure 2 a calibration plot of our model obtained by forming bins of similar prediction probabilities in the data. We can see that the model is well-calibrated, as the predicted probabilities broadly correspond to the ground truth probabilities in each bin.



(a) AUC by altitude



(b) AUC by watershed

Figure 3: Performance of our model across different geographical features.

2.2 Validity Domain

Analyse the validity of your model on the provided data and give recommendations on how the model should be used (strength and limitations) for the prediction of floods

Overall, the model presents a good performance profile across classification thresholds, and its ability to avoid false positives (see Section 1.2) means that a practitioner has the flexibility to set the desired level of false negative rate without creating too many false alarms. However, the 0-probabilities obtained on patches that we discarded from the training data do not indicate that a flood is completely impossible. Ideally, the training period should be extended and/or the choice of patches should be refined thanks to domain knowledge, especially on the occurence of extremely rare flooding events.

To further understand how the model behaves depending on where it is applied, we plot in Figure 3 the AUC scores at different levels of altitude (Figure 3a) and for different watersheds (Figure 3b). The performance clearly decreases for higher altitudes, which indicates the model is probably not adapted to forecasting floods in mountains. It is also likely that the floods are simply harder to predict in mountains as they are often more sudden than in the valleys. Interestingly, the performance also varies also watershed. The model performs best on the Seine basin, and struggles the most on the Somme basin. This result is quite surprising, especially as that area is largely flat and the poor performance cannot be explained by higher altitudes in that region.

3 Frugality

Explain how you minimized the computational footprint of your solution

Our filtering of training patches drastically reduces the amount of computations needed to train the model. Moreover, we prioritized when possible smaller networks that can yield strong results when tuned correctly but don't require as much computations for both training and inference.

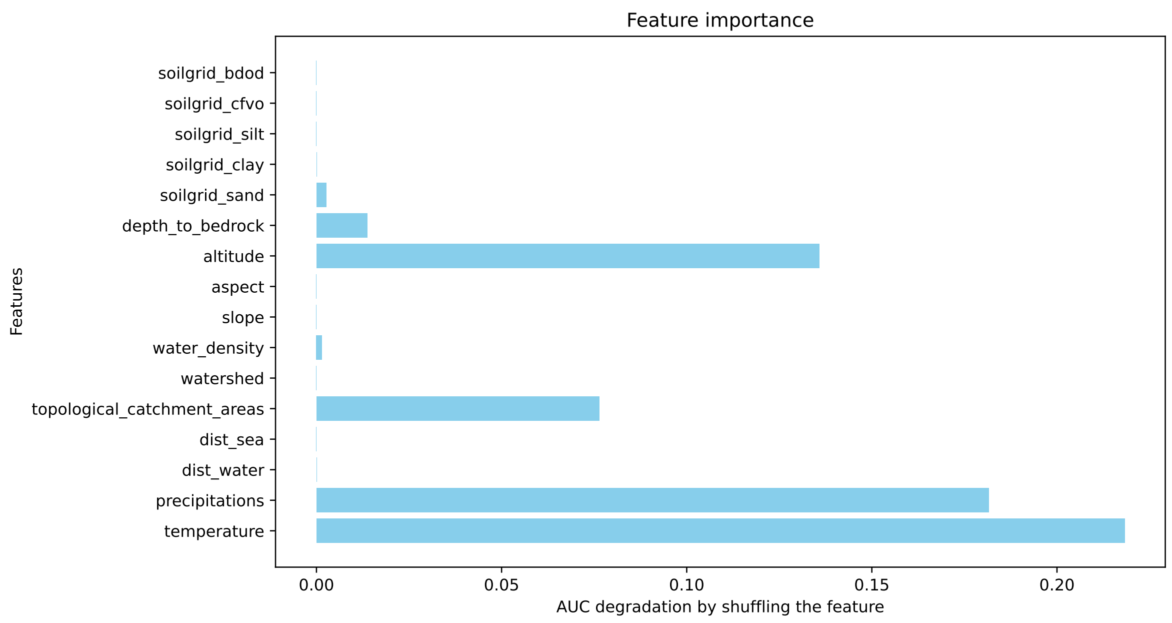


Figure 4: Permutation feature importance of our model.

4 Explainability

What features are important for your model and how do they influence predictions? Using explainability: evaluate the coherence of your model with your understanding of flooding mechanisms. Can you provide insights on flooding patterns and causes that could be useful for the user?

We performed a permutation importance analysis of our features, i.e., for each feature, we randomly shuffle the patches corresponding to the desired feature and recompute the ROC AUC. The degradation in performance represents feature importance. In Figure 4, we can see that spatio-temporal features (precipitations and temperature) carry the most importance among all features. Among geospatial features, altitude and the topological catchment area affect the AUC the most. This shows that our model is not static and is instead able to leverage the real-time spatio-temporal information.

Note that a feature with no reported importance can be a indicator of floods, but either our model was not able to extract the relevant information out of this feature, or the information it was able to extract is redundant with the information extracted from another feature. For instance, it is likely that our model does not use the watershed data to make its prediction as the topological catchment area feature contains the information it uses for the prediction. It is not surprising to us that the most distinctive features are spatio-temporal features, as they are known to have a large effect on flood events and cannot be accurately predicted from geospatial features only. We were at first more surprised by the importance of altitude compared to other geospatial features, but it is not completely unexpected as mountainous areas are known to show very specific topographical and climate patterns and can be subject to large floods in a short amount of time.

References

Varun Godbole, George E. Dahl, Justin Gilmer, Christopher J. Shallue, and Zachary Nado. Deep learning tuning playbook, 2023. URL http://github.com/google-research/tuning_playbook. Version 1.0.

Etienne Le Naour, Louis Serrano, Léon Migus, Yuan Yin, Ghislain Agoua, Nicolas Baskiotis, Patrick Gallinari, and Vincent Guigue. Time series continuous modeling for imputation and forecasting with implicit neural representations, 2023.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 12116–12128. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/652cf38361a209088302ba2b8b7f51e0-Paper.pdf.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4.