Towards AI for approximating hydrodynamic simulations as a 2D segmentation task

Lydia Bryan-Smith*[1] and Nina Dethlefs[2]

[1,2]University of Hull

1 Abstract

Traditional predictive simulations and remote sensing techniques for forecasting floods are based on fixed and spatially restricted physics-based models. These models are computationally expensive and can take many hours to run, resulting in predictions made based on outdated data. They are also spatially fixed, and unable to scale to unknown areas. By modelling the task as an image segmentation problem, an alternative approach using artificial intelligence to approximate the parameters of a physics-based model in 2D is demonstrated, enabling rapid predictions to be made in real-time.

2 Introduction

Predicting floods and other events caused by extreme weather relies on computational simulations using techniques such as cellular automata to predict them in advance [1]. These events can have severe impacts, such as destruction of property and injury and/or loss of life, making their prediction in a timely manner essential to avoid humanitarian disasters [2].

Physics-based simulations are used in a variety of contexts, including hydrology [3–5], climate [1], and others [6, 7] - but while accurate are computationally expensive and time consuming to run, resulting in predictions made using outdated data. This makes them unsuited for real-time estimation of hazardous phenomena in situations in which conditions are unpredictable.

AI has previously been used for predicting and monitoring such events in the form of simple ANNs [8], sequence to sequence (LSTM) [9], image classification [10–12], and more [13], but these approaches are limited to a single geographical point at a time (e.g. the flow rate of a river at a single point) - leaving characterisation of the flood itself to a hydrodynamic model (as explained in section 3.1 [9]).

In this paper, we present a feasibility study for approximating the parameters of a physics-based hydrological simulation [4] with an AI model. We achieve this by modelling the problem as an image semantic segmentation task. Our model, given an input of T time steps of input data that would nominally be fed to a physics-based model, predicts the output this computational model would have produced in real time.

While for our approach we utilise water depth prediction from rainfall radar data [14] and a heightmap [15], we aim that our approach be applicable to other domains. We make the following contributions in this paper:

- A state-of-the-art model architecture for predicting water depth in 2D
- The outline of a system could be used in harmony with existing hydrodynamic models to map floods in real time
- An illustration of improvements made from a baseline DeepLabV3+ model and their effectiveness in achieving accurate predictions, and the challenges that still remain

3 Related works

3.1 Computational simulations

Hydrological simulations to map and simulate the flow of water are available using a variety of algorithms. Models are either particle based [16] or cell based [4]. Models with 3 dimensions are the most computationally expensive, so are not used in this project. These are followed by models in 2D such as CAESAR-Lisflood [4] which have a lower computational cost.

Unfortunately, these models are still rather computationally expensive. Attempts have been made to reduce the computational requirements of these models. One option is to utilise GPU acceleration [5] or cellular automata [4], but while this does reduce the computational complexity it is not enough to run the model in real time over a large area.

Another avenue that has been explored to reduce this cost is simplifying the underlying algorithm. Such simplified algorithms [17, 18] are an order of magnitude less resource intensive, so as a result can process wider areas at once than e.g. 2D models implementing shallow water equations, but are less accurate [19].

At the other extreme, large climate simulations run by governments on high performance computers have the capability to simulate many parameters at once for large areas [1, 20], but are often commercial

*Corresponding Author: L.Bryan-Smith@hull.ac.uk

Proceedings of the 5th Northern Lights Deep Learning Conference (NLDL), PMLR 233, 2024.

and access to data is limited, aside from the clear computational cost barrier.

3.2 Learning Models

With the advent of Deep Learning, it has been used in a variety of contexts to predict floods. The simplest of these approaches is to predict water depth at a single point in a river a given amount of time in advance [9], or multiple points across a river basin [21]. This is only a proxy for predicting water depth across a wider area though, with a physics based model (section 3.1) required for this purpose.

Covering wider areas with AI has been tackled in a number of ways. Hybrid models involve using AI to make some prediction, and then hand off to a physics-based model [5, 22, 23]. Another option is to make and/or predict indirect measurements (e.g. the state of a municipal drainage system), and use these to predict the actual state of the world [24]. All of these options either have the same computational expense limitations as the hydrological models they call or make predictions for single geographical points, limiting generalisability.

Alternatively, data from satellites or automated drones can be used to estimate the extent of a flood [25–27]. These methods can produce accurate maps of the extent of a flooding event, but are either expensive and high effort (drones) or low frequency and sometimes limited by clouds (satellites).

4 Data

Before we can train a model (see section 5.1), a ground truth label upon which to train it must first be calculated. In our project, we achieve this by running the hydrological simulation model CAESAR-Lisflood [4] - specifically the C++ implementation HAIL-CAESAR [28], which we applied minor I/O format modifications to. The inputs to this model are geographically centred around the Humber estuary in the United Kingdom (as data from the UK was readily available, and the nature of the area is already known), though the model is designed to be trained on data from any location. These inputs are twofold:

1. **Rainfall radar data:** Every 5 minutes from 2006-01-01 to 2020-10-02, sized 105×174. Total timesteps: 1.48M [14].
2. **Heightmap:** Ordnance survey terrain 50 dataset [15]

The hydrological simulation model simulates water flow using simplified 2D shallow water equations, and outputs an absolute water depth map for the simulated area at the same time frequency as the input rainfall radar data, measured in metres. This map is then binarised into 2 classes: "water" (1) and "not water" (0) with a threshold of 0.1m. We calculate a ratio of 22.1%:77.9% (water:not water) between the 2 classes. This is the ground truth label used to train the model, as shown in figure 1. Data is stored in the tensorflow TFRecord format to optimise read performance. 4000 samples are stored per file, totalling 371 files.

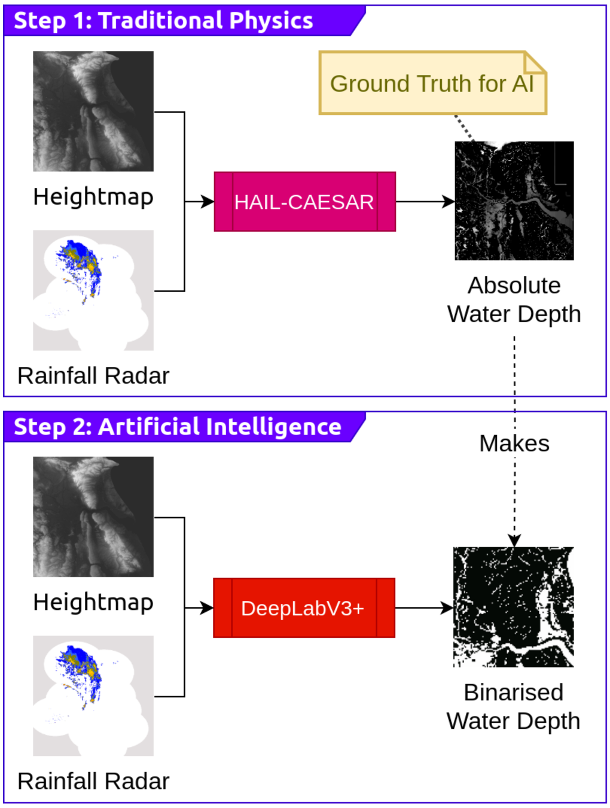

Figure 1. An overview of the process of by which the model was trained. First, ground truth labels are generated by running the rainfall radar and heightmap data through a traditional physics-based model [28]. Then, the resulting water depth data is binarised and used as a ground-truth for training a semantic segmentation AI model.

5 Approach

5.1 Semantic Segmentation

Image semantic segmentation models perform the task of pixel-wise classification on some input image, segmenting it into logically distinct categories. For example, an image of a typical urban street in the form of an $128 \times 128 \times 3$ channels-last matrix could be semantically segmented into categories such as `building`, `road`, and `person`, with the output in the form $128 \times 128 \times N$, where N is the number of classes to segment the input image into.

This technique has been adapted to make 2D predictions rainfall in the future, using rainfall radar data from a few hours prior [14, 29, 30]. While

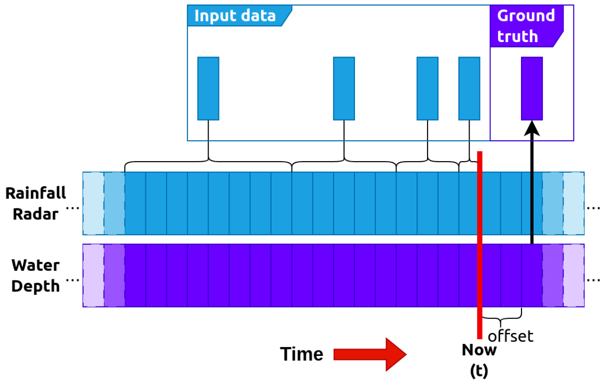

Figure 2. A visualisation of how data was preprocessed. t is moved forwards by one place at a time, and at each place input and ground truth data are calculated to make another sample the model can be trained on. Thus, a large number of time steps (96 in this case) are covered using a small amount of memory (see also section 5.2).

rainfall predictions are valuable, this is not the only source of flooding [31] and are of limited use for flooding prediction without being coupled to a hydrological simulation.

In this project, 'image' semantic segmentation is used to approximate water depth from rainfall radar data in 2D. Multiple semantic segmentation model architectures have been developed in the field, and such model architectures naturally lend themselves not only to their originally intended purpose (segmenting images), but also to our problem too. The earliest attempt the authors could find at a model architecture to solve the problem was FCN [32], which has a simple encoder followed by a 1-layer segmentation head. By far the most popular architecture for the task though is U-Net, which consists of a more full encoder-decoder autoencoder connected with skip connections [33]. SegNet is similar to U-Net, but improves on it by reducing memory usage (specifically by pooling directly before skip connections) [34], yet still falls short when handling complex scenes and minority classes.

Connectedness is a key theme of this class of model. PSPNet takes account of global structure with a pyramid architecture to attempt to handle the complexity and minority class issue [35], but the current state-of-the-art is DeepLabV3+ [36]. By combining a PSPNet-style pyramid with dilated (atrous) convolutions [37] and a U-Net encoder-decoder style model, significant performance improvements can be observed in both of the aforementioned areas. It is this model that forms the backbone of our approach.

5.2 Memory limits

A key challenge of training such a model on high-frequency data is GPU memory (VRAM) usage. To address this, we slice our data into samples and reduce the input into 8 channels of an 'image', as illustrated in figure 2. Consider two lists of $X \times Y$ matrices, W_t representing binarised ground truth water depth data from the hydrological simulation outlined in section 4, and R_t representing rainfall radar data.

The input to the model can be considered a sliding window. Set bin sizes $bs = [1, 3, 3, 5, 12, 24, 48]$, and then the rainfall radar data is processed as such:

$$R'_{t,c} = \overset{[}{i} = t - \sum_{j=0}^{c} bs] t - \sum_{j=0}^{c+1} bs \max|R_i| \qquad (1)$$

where each successive sublist of time steps collectively in the range $R_{[t-(\sum bs):t]}$ are collapsed into a single set of channels channel in the form $R'_{t,c}$ by taking the maximal value, and c is the channel of $R'_{t,c}$. This has the practical effect of reducing 96 channels to just 8, significantly reducing the VRAM required - and is represented visually in figure 2. As an addendum, the ground truth label the model learns from can be represented thus:

$$W'_t[x, y] = \\ (\sum W_{t+offset}[x-1 : x+1, y-1 : y+1]) \geq 1 \qquad (2)$$

where $offset$ is an offset to the timestep as described in section 4 and figure 2, and square-bracket notation $[x_a : x_b, y_a : y_b]$ indicates a submatrix. This effectively causes any cell of W_t with a value of 1 and all 8 neighbouring cells equal to 0 to be set to 0, removing isolated pixels.

Hence, intuitively the input to the model $R'_{t,c}$ is of shape $[batch, height, width, channel]$, and the ground truth W'_t is of shape $[batch, height, width]$, or $[batch, height, width, class]$ when one-hot encoded.

6 Experiments

Our model is based on DeepLabV3+ [36], which as explained in section 5.1 consists of an image encoder (ResNet50), followed by a pyramid architecture such that the 2D output is a 'semantic segmentation' with each pixel being the binary classes water/no water. In other words, a per-pixel probability of each class - which is identical in shape to the ground truth labels (W'_t) used for training (all one-hot encoded). To decode the one-hot encoded output, for each pixel the highest value is taken to be the predicted class (e.g. $P'_{x,y} = f(P_{x,y,c})$, if c is the class) - as in $P'[x, y] = i$ **of** $\max|P[x, y]_i|$, where i is the class index (i.e. in this case $i \in \{0, 1\}$).

The input to this model is the rainfall radar data and the heightmap, as in section 4 - these are combined so that the heightmap becomes a channel of $R'_{t,c}$. Our model has following hyperparameters:

- **Total parameters:** 11.87M

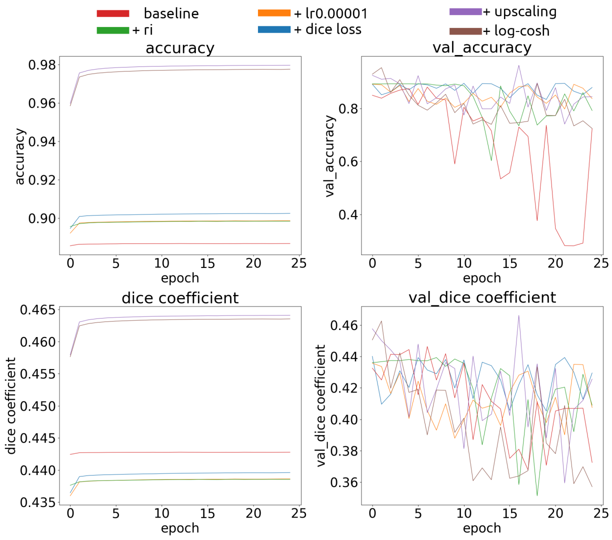

Figure 3. Accuracy and Dice loss plots for the models we trained. Overfitting is observed if models are trained for too long. All adjustments (see also table 1) show improvements apart from log-cosh, which was dropped.

- **Learning rate:** 0.00001
- **Batch size:** 32
- **Encoder:** ResNet50 [38]
- **Loss function:** Additive cross-entropy and Dice loss (i.e. $loss = cel + dice$)
- **Upscaling:** We adjusted to the original DeepLabV3+ model [37] to scale the input from 128×128 to 256×256 using `nearest` interpolation
- **Offset:** From rainfall radar data, water depth 5 minutes in the future is predicted.

We chose these hyperparameters based on a series of experimental comparisons. We trained our models on various Nvidia GPUs, subject to availability: Nvidia (A40), Tesla (K40m, P100), GeForce (2060, 3060). Models trained for a total of 25 epochs, and we picked the checkpoint with the highest validation accuracy. We used an 80% - 20% training-validation data split, with files being randomly allocated to each using the fisher-yates shuffling algorithm [39]. All models were trained with the same files in the same partition of the split.

With these settings, our model took a total of 3 days 7 hours to train, and 36 seconds to make a prediction (processing an entire batch at a time). The source 2D hydrological model that generated the water depth label data took 14 days 8 hours to complete. These include setup and teardown, such as loading system libraries.

The metrics from the models we trained are presented in table 1. Starting from an unmodified baseline DeepLabV3+ model, we iteratively improved the model to function more effectively on our task. We measured a number of metrics including accuracy ('acc'), dice coefficient ('dice coef'), and intersection-over-union ('mean iou'). We show plots of the former two in figure 3. Performance gains are obtained relatively early in the training process, with further epochs leading to overfitting.

6.1 Discussion

The biggest issue we faced with our learning task was the output resolution of the semantic segmentation model - especially on the boundary between classes. Adding **remove_isolated**() as described in section 4 (+1.3% val acc.), changing our loss function from simple cross-entropy-loss to include dice loss additively (+0.4%), and upscaling the input (x2; +6.8%) all improved validation accuracy (shown in brackets). We also experimented with using a loss function of $loss = cel + \log(\cosh(dice))$ as [40] suggests that $\log(\cosh(dice))$ is more tractable than simply *dice*, but we found it to reduce our validation accuracy and all other metrics (-0.8%). We reduced the learning rate from the default of 0.001 to 0.00001, as we observed this reduced instability in validation accuracy and the dice coefficient - though this reduced our accuracy slightly (-0.2%). The dataset is also very large (about 1.4M samples), further supporting the reduction in learning rate. All metrics are validation accuracy.

With this in mind, we pick the hyperparameters of the model with upscaling but not log-cosh. We plot predictions from each model alongside ground truth in figure 4. The adjusted model is shown to make predictions at a higher fidelity than the initial baseline, effectively matching the patterns in the ground truth by making predictions based on the input rainfall and heightmap information.

A key limitation in designing the model was visual fidelity and GPU video memory (VRAM) as described in section 5.2. Much of the design was done before the Nvidia A40 mentioned above was available (which has 48GB VRAM), limiting our model design choices to those that fit within a 16 GB VRAM envelope. With additional computational resources now available, the visual fidelity may potentially be further improved at the expense of more VRAM through additional upscaling - perhaps by x8 compared to the x2 used in the models presented here.

Another observation we make is of significant variation in validation accuracy during the training process. This is emphasised if we calculate the standard deviation of the metrics (excluding the first five epochs) as reported in table 2. We suggest that this is due to the complexity of the task and the predicted output. The complexity of the task could be reduced by adjusting the source hydrological simulation used. For example, this could include filling in holes in the heightmap, or choosing a different hydrological simulation model that

	acc	**dice coef**	**mean iou**	**val acc**	**val dice coef**	**val mean iou**
baseline	0.887	0.443	0.747	0.880	0.446	0.737
+ ri	0.897	0.438	0.767	0.893	0.437	0.759
+ lr0.00001	0.892	0.436	0.756	0.891	0.436	0.754
+ dice loss	0.902	0.439	0.776	0.895	0.438	0.762
+ upscaling	**0.979**	**0.464**	**0.949**	**0.963**	**0.466**	**0.911**
+ log-cosh	0.974	0.464	0.935	0.955	0.463	0.890

Table 1. Metrics for our DeepLabV3+-based binarised water depth prediction models. For each model variant, the model was trained for 25 epochs, and then the epoch with the highest validation accuracy was chosen for display here. We iteratively improved on a baseline DeepLabV3+ model [36] by removing positive pixels surrounded by negative ones (ri), reducing the learning rate (lr0.00001), including dice loss additively (dice loss), upscaling the inputs and outputs (upscaling), and adding log(cosh($dice\_loss$)) (log-cosh) - though the latter adjustment slightly reduced performance, so was discarded.

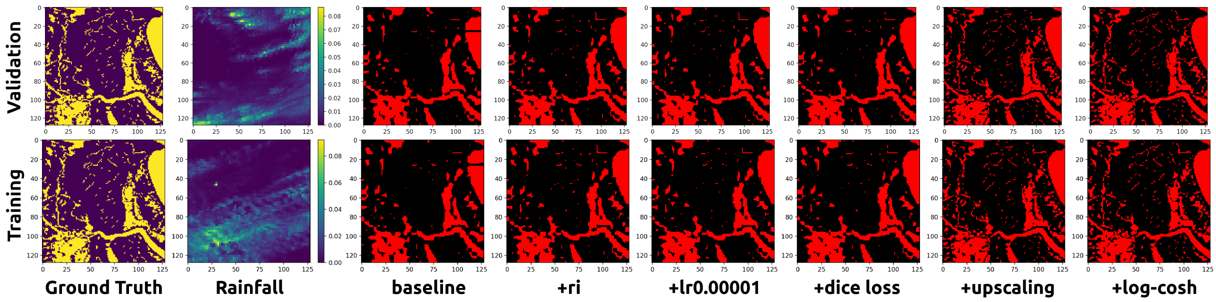

Figure 4. Training predictions by each of the models described in table 1, along with the ground truth prediction (left) and the processed rainfall data that is used as input (1 from the left). When reading digitally, we recommend zooming in to observe the at times subtle differences between the different predictions.

accounts for groundwater flow. By implementing these changes, we would expect that the complexity of the prediction task and the variation in validation metrics would be lowered, and the quality of predictions improved.

We observe a general negative trend in model validation performance the more epochs it is trained for. This suggests that our model is overfitting if trained for too long. We also note that, as mentioned in section 4, our dataset is somewhat unbalanced. We used additive dice to our loss function to counteract this, but exploration of other techniques like weighted and shape-aware loss functions for example would be worthwhile.

Our experiments demonstrate the feasibility of our approach to predict floods in two dimensions. The accuracy is limited by the hydrological simulation the model is trained on, but with some adjustments to the hydrological simulation as explained above our approach could effectively and rapidly make direct predictions of water depth. We anticipate that, after proper characterisation and analysis, the model may be able to make accurate predictions up to a few hours in advance after making these improvements.

	Training	Validation
acc	2.97×10^{-4}	0.0513
dice coef	8.19×10^{-5}	0.0249
mean iou	7.27×10^{-4}	0.117

Table 2. Standard deviation values for the `+ upscaling` model chosen in section 6.1 and presented in table 1. Significant variation was observed in validation metrics.

7 Conclusion

Using the DeepLabV3+ model [36] as a base, we have demonstrated a proof of concept for a new method of directly predicting floods from rainfall radar data and a heightmap in two dimensions. We accomplish this by approximating a hydrological simulation model. While our model's performance is limited by the nature and accuracy of the simulation chosen (given that to the best of the authors' knowledge no real-world dataset with a sufficient temporal resolution exists to serve as training labels), our method predicts an entire area at once - avoiding the need for models to be retrained and maintained for many individual locations or being combined with an expensive hydrological simulation to make useful multidimensional predictions.

In addition, our approach makes predictions more directly than previous approaches that rely on e.g. static camera footage [41] river levels [9] or secondary sensor networks [24].

Such advance warning is essential for minimising humanitarian risk and preventing loss of life. To develop this proof of concept further, we want to more fully characterise the model's strengths and weaknesses - e.g. under different weather patterns. Future work could also include tuning the source hydrological simulation based on the lessons learned about the nature of the semantic segmentation task framing to improve performance, and accounting for additional variables in our simulation such as sources of flooding other than rainfall (e.g. tidal), groundwater flow, and municipal drainage systems.

We also want to investigate developing a geographically-invariant version of the model that can use e.g. a tiled approach to make predictions for larger areas without the need for retraining the model, as the current version cannot generalise to other geographical areas without retraining. A tiled approach would split a rainfall/heightmap/water depth dataset into equal squared tiles before training to make predictions, and could also increase the resolution of predictions with a suitable hydrological model as an input.

Finally, increasing the fidelity of predictions to 3 or more classes beyond binarised "water"/"no water" would also improve usefulness of predictions made.

Ultimately, a range of approaches is required to effectively address the problem of accurately forecasting floods and their extent in real time. This includes not only hydrodynamic / climatological simulations [1, 4] and our approach to approximating them with AI, also other diverse data sources such as satellite data [42], remote sensing [24], and the analysis of human-centred approaches like crowdsourcing [43] and social media [44, 45]. Different sources complement each other and can improve visibility / situational awareness.

7.1 Acknowledgments

Lydia Bryan-Smith is funded by a PhD stipend from the University of Hull. We acknowledge the VIPER high-performance computing facility of the University of Hull and its support team.

References

[1] H. Hersbach, B. Bell, P. Berrisford, S. Hirahara, A. Horányi, J. Muñoz-Sabater, J. Nicolas, C. Peubey, R. Radu, D. Schepers, A. Simmons, C. Soci, S. Abdalla, X. Abellan, G. Balsamo, P. Bechtold, G. Biavati, J.-R. Bidlot, M. Bonavita, G. D. Chiara, P. Dahlgren, D. Dee, M. Diamantakis, R. Dragani, J. Flemming, R. G. Forbes, M. Fuentes, A. J. Geer, L. Haimberger, S. B. Healy, R. J. Hogan, E. V. Holm, M. Janisková, S. P. E. Keeley, P. Laloyaux, P. Lopez, C. Lupu, G. Radnoti, P. de Rosnay, I. Rozum, F. Vamborg, S. Villaume, and J.-N. Thepaut. "The ERA5 global reanalysis". In: *Quarterly Journal of the Royal Meteorological Society* 146 (2020), pp. 1999–2049.

[2] S. A. Vicario, M. C. Mazzoleni, S. Bhamidipati, M. Gharesifard, E. Ridolfi, C. Pandolfo, and L. Alfonso. "Unravelling the influence of human behaviour on reducing casualties during flood evacuation". In: *Hydrological Sciences Journal* 65 (2020), pp. 2359–2375.

[3] Z. Shu. "Introduction of Numerical Simulation Software Dleft3D & Its Application on Offshore Area of Aojiang Estuarine". In: *Journal of China Hydrology* (2007).

[4] T. J. Coulthard, J. C. Neal, P. D. Bates, J. Ramirez, G. A. M. de Almeida, and G. R. Hancock. "Integrating the LISFLOOD-FP 2D hydrodynamic model with the CAESAR model: implications for modelling landscape evolution". In: *Earth Surface Processes and Landforms* 38.15 (2013), pp. 1897–1906.

[5] L. Smith, Q. Liang, P. James, and W. Lin. "Assessing the utility of social media as a data source for flood risk management using a real-time modelling framework". In: *Journal of Flood Risk Management* 10.3 (2017), pp. 370–380.

[6] L. Casalino, A. C. Dommer, Z. Gaieb, E. P. Barros, T. Sztain, S.-H. Ahn, A. Trifan, A. Brace, A. T. Bogetti, A. R. Clyde, H. Ma, H. Lee, M. Turilli, S. Khalid, L. T. Chong, C. Simmerling, D. J. Hardy, J. D. C. Maia, J. C. Phillips, T. Kurth, A. Stern, L. Huang, J. D. McCalpin, M. Tatineni, T. Gibbs, J. E. Stone, S. Jha, A. Ramanathan, and R. E. Amaro. "AI-driven multiscale simulations illuminate mechanisms of SARS-CoV-2 spike dynamics". In: *The International Journal of High Performance Computing Applications* 35 (2021), pp. 432–451.

[7] J. A. Harris, R. Chu, S. M. Couch, A. Dubey, E. Endeve, A. Georgiadou, R. Jain, D. Kasen, M. P. Laiu, O. E. B. Messer, J. O'Neal, M. A. Sandoval, and K. Weide. "Exascale models of stellar explosions: Quintessential multi-physics simulation". In: *The International Journal of High Performance Computing Applications* 36 (2021), pp. 59–77.

[8] G. Furquim, G. P. R. Filho, R. Jalali, G. Pessin, R. W. Pazzi, and J. Ueyama. "How to Improve Fault Tolerance in Disaster Predictions: A Case Study about Flash Floods

Using IoT, ML and Real Data." In: *Sensors* 18.3 (2018), p. 907.

[9] X.-H. Le, H. V. Ho, G. Lee, and S. Jung. "Application of Long Short-Term Memory (LSTM) Neural Network for Flood Forecasting". In: *Water* 11.7 (2019), p. 1387.

[10] R.-Q. Wang, H. Mao, Y. Wang, C. Rae, and W. Shaw. "Hyper-resolution monitoring of urban flooding with social media and crowdsourcing data". In: *Computers & Geosciences* 111 (Feb. 2018), pp. 139–147. DOI: 10.1016/j.cageo.2017.11.008. URL: https://doi.org/10.1016%2Fj.cageo.2017.11.008.

[11] B. K. Mishra, D. Thakker, S. Mazumdar, D. Neagu, M. Gheorghe, and S. Simpson. "A novel application of deep learning with image cropping: a smart city use case for flood monitoring". In: *Journal of Reliable Intelligent Environments* 6 (2020), pp. 51–61.

[12] H. Ning, Z. Li, M. E. Hodgson, and C. Wang. "Prototyping a Social Media Flooding Photo Screening System Based on Deep Learning". In: *ISPRS international journal of geo-information* 9.2 (2020), p. 104.

[13] M. M. de Vitry, S. Kramer, J. D. Wegner, and J. P. Leitão. "Scalable flood level trend monitoring with surveillance cameras using a deep convolutional neural network". In: *Hydrology and Earth System Sciences* 23.11 (2019), pp. 4621–4634.

[14] The United Kingdom Meteorological Office and NCAS British Atmospheric Data Centre. *1 km Resolution UK Composite Rainfall Data from the Met Office Nimrod System*. Available online: https://catalogue.ceda.ac.uk/uuid/27dd6ffba67f667a18c62de5c3456350 [Accessed 20/07/2023]. 2003.

[15] Ordnance Survey. *Terrain 50 — Visualise and Analyse Landscapes in 3D*. Available online: https://www.ordnancesurvey.co.uk/business-government/products/terrain-50 [Accessed 09/04/2020]. 2020.

[16] Deltares. *Home - DLEFT3D*. Available online: https://oss.deltares.nl/web/delft3d [Accessed 12/07/2022]. 2021.

[17] J. Lhomme, P. B. Sayers, B. P. Gouldby, P. G. Samuels, M. Wills, and J. Mulet-Marti. "Recent development and application of a rapid flood spreading method". In: *FLOODrisk2008 - Flood Risk Management: Research and Practice*. 2008.

[18] A. D. Nobre, L. A. Cuartas, M. Hodnett, C. D. Rennó, G. Rodrigues, A. Silveira, M. Waterloo, and S. Saleska. "Height Above the Nearest Drainage - a hydrologically relevant new terrain model". In: *Journal of Hydrology* 404.404 (2011), pp. 13–29.

[19] J. Teng, A. Jakeman, J. Vaze, B. Croke, D. Dutta, and S. Kim. "Flood inundation modelling". In: *Environmental Modelling and Software* 90.90 (2017), pp. 201–216.

[20] G. Zängl, D. Reinert, P. Rípodas, and M. A. Baldauf. "The ICON (ICOsahedral Non-hydrostatic) modelling framework of DWD and MPI-M: Description of the non-hydrostatic dynamical core". In: *Quarterly Journal of the Royal Meteorological Society* 141 (2015).

[21] Z. Moshe, A. Metzger, G. Elidan, F. Kratzert, S. Nevo, and R. El-Yaniv. "HydroNets: Leveraging River Structure for Hydrologic Modeling". In: *EGU General Assembly Conference Abstracts*. EGU General Assembly Conference Abstracts. May 2020, 4135, p. 4135.

[22] H. Mojaddadi, B. Pradhan, H. Nampak, N. Ahmad, and A. H. Ghazali. "Ensemble machine-learning-based geospatial approach for flood risk assessment using multi-sensor remote-sensing data and GIS". In: *Geomatics, Natural Hazards and Risk* 8 (2017), pp. 1080–1102.

[23] S. Nevo, V. Anisimov, G. Elidan, R. El-Yaniv, P. Giencke, Y. Gigi, A. Hassidim, Z. Moshe, M. Schlesinger, G. Shalev, A. Tirumali, A. Wiesel, O. Zlydenko, and Y. Matias. "ML for Flood Forecasting at Scale." In: *Proceedings of the NIPS AI for Social Good Workshop*. 2018.

[24] K. L. Keung, C. K. M. Lee, K. K. H. Ng, and C. K. Yeung. "Smart City Application and Analysis: Real-time Urban Drainage Monitoring by IoT Sensors: A Case Study of Hong Kong". In: *2018 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. 2018, pp. 521–525.

[25] G. Panteras and G. Cervone. "Enhancing the temporal resolution of satellite-based flood extent generation using crowdsourced data for disaster monitoring". In: *International Journal of Remote Sensing* 39 (2016), pp. 1459–1474.

[26] B. Peng, Z. Meng, Q. Huang, and C. Wang. "Patch Similarity Convolutional Neural Network for Urban Flood Extent Mapping Using Bi-Temporal Satellite Multispectral Imagery". In: *Remote. Sens.* 11 (2019), p. 2492.

[27] A. Sharma and U. Verma. "Flood Magnitude Assessment from UAV Aerial Videos Based on Image Segmentation and Similarity". In: *TENCON 2021 - 2021 IEEE Region 10 Conference (TENCON)*. 2021, pp. 476–481. DOI: 10.1109/TENCON54134.2021.9707250.

[28] Declan Valters. *GitHub - dvalters/HAIL-CAESAR: The High-Performance Architecture-Independent LISFLOOD-CAESAR model of floodplain, river, and sediment dynamics*. Available online: https://github.com/dvalters/HAIL-CAESAR [Accessed 07/04/2020]. 2018.

[29] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo. "Convolutional LSTM Network: a machine learning approach for precipitation nowcasting". In: *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*. 2015, pp. 802–810.

[30] C. L. Bromberg, C. Gazen, J. J. Hickey, J. Burge, L. Barrington, and S. Agrawal. "Machine Learning for Precipitation Nowcasting from Radar Images". In: *Machine Learning and the Physical Sciences Workshop at the 33rd Conference on Neural Information Processing Systems (NeurIPS)*. 2019, p. 4.

[31] S. N. Jonkman. "Global Perspectives on Loss of Human Life Caused by Floods". In: *Natural Hazards* 34 (2005), pp. 151–175.

[32] E. Shelhamer, J. Long, and T. Darrell. "Fully convolutional networks for semantic segmentation". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2014), pp. 3431–3440.

[33] 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*. 2015, pp. 234–241.

[34] V. Badrinarayanan, A. Kendall, and R. Cipolla. "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (2015), pp. 2481–2495.

[35] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. "Pyramid Scene Parsing Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 6230–6239.

[36] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *European Conference on Computer Vision*. 2018.

[37] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam. "Rethinking Atrous Convolution for Semantic Image Segmentation". In: *ArXiv* abs/1706.05587 (2017). DOI: 10.48550/arXiv.1706.05587.

[38] K. He, X. Zhang, S. Ren, and J. Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[39] M. Eberl. "Fisher–Yates shuffle". In: *Archive of Formal Proofs* (Sept. 2016). https://isa-afp.org/entries/Fisher_Yates.html, Formal proof development. ISSN: 2150-914x.

[40] S. Jadon. "A survey of loss functions for semantic segmentation". In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)* (2020), pp. 1–7.

[41] K. Lohumi and S. Roy. "Automatic Detection of Flood Severity Level from Flood Videos using Deep Learning Models". In: *2018 5th International Conference on Information and Communication Technologies for Disaster Management (ICT-DM)*. 2018, pp. 1–7.

[42] M. O. Sghaier, M. Hadzagic, and J. Patera. "Fusion of SAR and Multispectral Satellite Images Using Multiscale Analysis and Dempster-Shafer Theory for Flood Extent Extraction". In: *2019 22th International Conference on Information Fusion (FUSION)* (2019), pp. 1–8. DOI: 10.23919/fusion43075.2019.9011209. URL: https://doi.org/10.23919/fusion43075.2019.9011209.

[43] N. Kankanamge, T. Yigitcanlar, A. Goonetilleke, and M. Kamruzzaman. "Determining disaster severity through social media analysis: Testing the methodology with South East Queensland Flood tweets". In: *International journal of disaster risk reduction* 42 (2020), p. 101360.

[44] A. Kongthon, C. Haruechaiyasak, J. Pailai, and S. Kongyoung. "The role of Twitter during a natural disaster: Case study of 2011 Thai Flood". In: *2012 Proceedings of PICMET '12: Technology Management for Emerging Technologies*. 2012, pp. 2227–2232.

[45] L. Bryan-Smith, J. Godsall, F. George, K. Egode, N. Dethlefs, and D. Parsons. "Real-time social media sentiment analysis for rapid impact assessment of floods". In: *Computers & Geosciences* (2023), p. 105405. ISSN: 0098-3004. DOI: https://doi.org/10.1016/j.cageo.2023.105405. URL: https://www.sciencedirect.com/science/article/pii/S0098300423001097.