International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

Satellite Image-Based Flooded Road Detection Using Semantic Segmentation

B. Spandana[1], Vidushi Bahuguna[2], Abhinaya Kanuri[3], Sai Supreetha Kusuma[4], Asmita Reddy Vangala[5]
[1]Assistant Professor, Computer Science Engineering (Artificial Intelligence & Machine Learning), G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana, India (500104)
[2,3,4,5]UG - Computer Science Engineering (Artificial Intelligence & Machine Learning), G. Narayanamma Institute of Technology and Science, Hyderabad, Telangana, India (500104)
Emails: *spandana.d@gnits.ac.in[1], to.vidushibahuguna@gmail.com[2], abhinayakanuri12@gmail.com[3], saisupreethakusuma@gmail.com[4], asmitareddy2135@gmail.com[5]*

Abstract
During floods, submerged roads cut off rescue teams from affected people, yet most satellite flood products map general inundation and do not say which roads can still be used. This work presents a deep learning pipeline that identifies accessible routes during floods from satellite imagery. A U-Net with an ImageNet-pretrained ResNet34 encoder takes co-registered pre-event and post-event four-band tiles from the SpaceNet 8 Germany subset and predicts a road map and a flooded-road map directly. Training combines weighted binary cross-entropy with Dice loss to counter severe class imbalance, and tiles are split into training, validation and test sets by geographic block to avoid spatial leakage. The predicted road mask is skeletonised into a road graph, edges with submerged stretches are removed, and a shortest-path search returns the fastest remaining route between two locations, or reports that none exists. On the held-out test set the model reaches a flooded-road F1 score of 0.74 and a road F1 score of 0.79. In 1,000 sampled origin–destination queries, 93% of the returned routes avoid every truly flooded segment, a route is found for 95% of reachable destinations, and the mean detour is 9% longer than the ideal flood-aware route. Using pre-event imagery and a Dice term raises the flooded-road F1 score from 0.42 to 0.74 over a post-event, cross-entropy baseline. The results show that flood-aware routing from satellite imagery alone is practical for emergency planning.
Keywords: *Deep learning; Flood mapping; Road network analysis; Route planning; Satellite imagery*

1. Introduction
Floods are among the most frequent and damaging natural disasters, and the first hours after onset largely decide how many people can be reached and helped. In these hours the limiting factor is often not the availability of rescue resources but the ability to reach people: roads become submerged, and routing decisions based on outdated maps can send vehicles into water that cannot be crossed. Responders therefore need to know not only where the water is, but which roads can still be used and which route to take. Satellite earth observation provides rapid, wide-area coverage after a flood, and deep learning has become the dominant approach for extracting information from such imagery. Flood extent mapping and road extraction have each been studied extensively (Long et al., 2015; Zhang et al., 2018; Bonafilia et al., 2020), but they are usually treated separately: flood maps describe inundation without reference to infrastructure, and road maps describe the network without reference to its current state. Even where flooded roads are detected, the output is a pixel mask and not a decision. A rescue coordinator still has to work out by hand which routes remain open. This paper closes that gap. We propose a pipeline that goes from raw pre-event and post-event satellite tiles to an accessible route: a single segmentation network detects roads and flooded roads, the predicted road mask is converted into a road graph, flooded segments are removed, and a shortest-path search returns the fastest remaining route between two locations or reports that none exists.

1.1.Related work
Fully convolutional networks (Long et al., 2015) established dense pixel-wise prediction, and the U-Net (Ronneberger et al., 2015) added a symmetric encoder–decoder with skip connections that recover fine spatial detail. Replacing the plain encoder with a

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

residual network (He et al., 2016) pretrained on ImageNet (Deng et al., 2009) improves convergence when labelled data are scarce, which is the case for flood imagery. Road extraction is difficult because roads are thin, long and often occluded by trees, shadows and buildings. Residual U-Nets give strong results with few parameters (Zhang et al., 2018), and the SpaceNet series (Van Etten et al., 2018) provides large benchmarks for road network extraction. Later work converts predicted road masks into graphs with travel-time estimates (Van Etten, 2020), which is the representation needed for routing, but these works target the static network and do not model its condition after a disaster. Datasets such as Sen1Floods11 (Bonafilia et al., 2020) and FloodNet (Rahnemoonfar et al., 2021) support flood extent mapping and scene understanding, and xBD (Gupta et al., 2019) addresses building damage. Change detection networks that compare images before and after an event, for example fully convolutional Siamese networks (Daudt et al., 2018), are a natural fit for disaster analysis. SpaceNet 8 (Hänsch et al., 2022) is notable in combining roads, buildings and flood state in one benchmark with paired pre-event and post-event tiles, and it is the dataset used here. Work built on this benchmark has concentrated on segmentation accuracy; turning flooded-road predictions into route recommendations has received less attention.

1.2.Objectives and contributions

The objective of this work is to identify, from satellite imagery alone, which roads remain accessible during a flood and which routes can be used between two locations. The contributions are: (i) a mask-generation pipeline that converts the SpaceNet 8 reference CSV, which stores pixel-space road geometries, into aligned road and flooded-road masks at native resolution; (ii) a U-Net with a ResNet34 encoder that uses pre-event and post-event imagery, including the near-infrared band, and predicts flooded roads directly with a combined cross-entropy and Dice loss; (iii) a graph-based routing stage that marks road segments as blocked or open and returns flood-aware routes; and (iv) an evaluation protocol with a geographically separated test set that reports both pixel-level segmentation metrics and route-level metrics.

2. Method

Figure 1 shows the complete system. It has two stages: a segmentation stage that predicts road and flooded-road maps from imagery, and a routing stage that converts these maps into a road graph and computes accessible routes. As shown in Figure 1 Overview of the proposed flood-aware route identification pipeline.

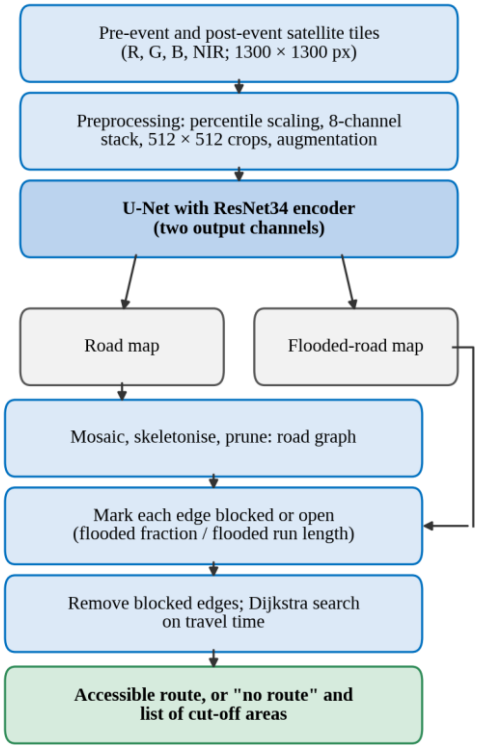


Figure 1 Overview of the proposed flood-aware route identification pipeline

2.1.Dataset

We use the Germany subset of SpaceNet 8 (Hänsch et al., 2022). The raw data comprise post-event multispectral GeoTIFF tiles of 1300 × 1300 pixels with four bands (red, green, blue and near-infrared), pre-event tiles of the same locations in the same format, and a reference CSV file with the ground-truth annotations. Each tile is named <acquisitionID>_<prefix>_<col>_<row>.tif, with column and row indices in the ranges 15–45 and 56–70. Post-event tiles come from two satellite acquisitions and pre-event tiles from one. In total 282 labelled tiles cover a grid of roughly 30 × 15 tiles.

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

Table 1 summarises the columns of the reference CSV used here. Each row describes one road segment as a LINESTRING in pixel coordinates, with a Boolean flag that states whether the segment is submerged, its length in metres and an estimated travel time. As shown in Table 1 Reference CSV schema.

Table 1 Reference CSV schema

Column	Description
ImageId	Tile identifier
Object	Feature type (road or building)
Wkt_Pix	Road geometry (LINESTRING, pixels)
Flooded	Whether the segment is submerged
length_m	Segment length (m)
travel_time_s	Estimated travel time (s)

2.2.Preprocessing and mask generation

Because the CSV stores geometry in pixel coordinates, masks are produced by direct rasterisation without geographic reprojection. Road-segment rows are grouped by a grid key (the <prefix>_<col>_<row> suffix of the tile name), which decouples annotations from the acquisition identifier, and each post-event tile is paired with its pre-event counterpart through the same key. Every LINESTRING is parsed and drawn with cv2.polylines at the native 1300-pixel resolution with a width of 5 pixels to form the road mask; segments with Flooded = True are drawn on a second canvas to form the flooded-road mask. The flooded-road mask is therefore a strict subset of the road mask, and the remaining road pixels form the dry-road mask. Masks are stored as 8-bit PNG files, and a manifest records the paths of every tile, its pre-event tile and its masks. Images are read with rasterio and all four bands of both dates are kept. Each band is clipped at its 2nd and 98th percentile and rescaled, which compresses the 16-bit dynamic range without letting outliers dominate. The two dates are stacked into one tensor with eight channels (pre-event R, G, B, NIR followed by post-event R, G, B, NIR), and each channel is standardised with the mean and standard deviation of the training set. Training uses random crops of 512 × 512 pixels taken at native resolution, so thin roads are not blurred by downscaling. Half of the crops are drawn so that they contain flooded road whenever the tile has any, which raises the share of positive pixels that the network sees. At inference, full tiles are processed with overlapping windows and blended [1 – 10].

2.3.Data augmentation

Augmentation is implemented with Albumentations (Buslaev et al., 2020). The same geometric transformation is applied to the pre-event image, the post-event image and both masks, so that the pair stays aligned. Photometric changes are drawn independently for each date, which imitates differences in illumination and sensor between two acquisitions. The validation and test transforms contain only normalisation. As shown in Table 2 lists the training transforms.

Table 2 Training augmentation pipeline

Transform	Parameters
Horizontal / vertical flip	p = 0.5 each
RandomRotate90	p = 0.5
Affine	shift ±5%, scale 0.9–1.1, rot. ±15°, p = 0.4
Brightness / contrast	±0.2, p = 0.5 (per date)
RandomGamma	80–120, p = 0.3 (per date)
Gaussian noise	p = 0.2
CoarseDropout	1–4 holes, 16–32 px, p = 0.15

2.4.Network architecture

The model is a U-Net (Ronneberger et al., 2015) with a ResNet34 encoder (He et al., 2016) initialised from ImageNet weights (Deng et al., 2009) and built with the segmentation-models-pytorch library (Iakubovskii, 2019). It has about 24 million parameters. The first convolution is extended from three to eight input channels; the pretrained filters are replicated across the extra channels and rescaled, so that the early layers start from useful weights. Pre-event and post-event bands are fused at the input (early fusion), which lets the first layers compare the two dates directly. The encoder produces features at five scales, each passed through a skip connection to the matching decoder stage, and a final 1 × 1 convolution maps the decoder output to two logit

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

channels: channel 0 for road and channel 1 for flooded road As shown in (Figure 2).

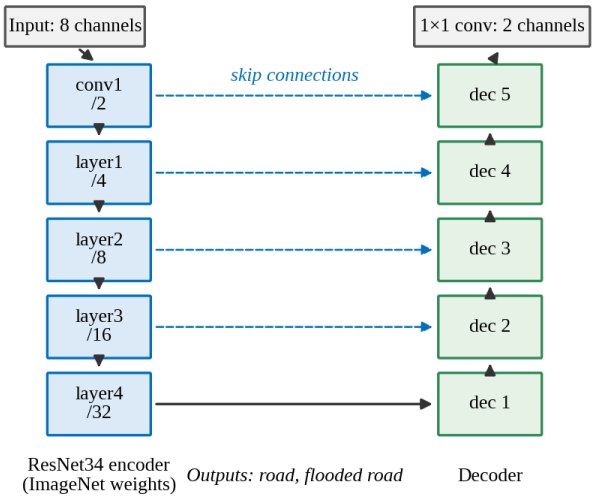


Figure 2 U-Net with ResNet34 encoder and two output channels

In contrast to a design in which the flooded-road map is obtained afterwards as the intersection of a road map and a flood map, here the flooded-road map is a direct output of the network. Because flood labels exist only on road segments, this output learns to separate flooded road from dry road, and it avoids the compounding of errors that occurs when two independent predictions are multiplied. Predicted flooded-road components smaller than 40 pixels are removed to suppress isolated false detections.

2.5.Loss function

Both outputs are trained with a combination of weighted binary cross-entropy and Dice loss:

$$L = \lambda_f L_f + \lambda_r L_r \quad (1)$$

$$L_k = BCE_w(z_k, y_k) + [1 - Dice(\sigma(z_k), y_k)] \quad (2)$$

where Lk is the loss of output k, z denotes the raw logits, y the binary ground truth, σ the sigmoid, and k ∈ {f, r} indexes the flooded-road and road outputs. The weighted BCE uses a positive-class weight equal to the ratio of negative to positive pixels in the training set, capped at 30. Road and flooded-road pixels occupy a very small fraction of an aerial tile, and plain cross-entropy can be minimised by predicting background everywhere. The positive-class weight and the overlap-based Dice term (Milletari et al., 2016) keep the gradient focused on the rare positive class; focal loss (Lin et al., 2017) is a common alternative. We set $\lambda f = 1.0$ and $\lambda r = 0.5$, because the flooded-road output is the one that decides routing [11 – 20].

2.6.Road graph construction

Predictions of neighbouring tiles are stitched into a mosaic using the grid key. The road probability map is thresholded (tr = 0.5), small gaps are closed with a 3 × 3 morphological closing, and the mask is skeletonised with a thinning algorithm (Zhang & Suen, 1984) from scikit-image (van der Walt et al., 2014). Spurs shorter than 10 pixels are pruned. Graph nodes are placed at junctions and end points, and edges are the pixel paths between them, simplified with the Douglas–Peucker algorithm (Douglas & Peucker, 1973) (ε = 2 px). Edge length is the path length multiplied by the ground sampling distance read from the GeoTIFF metadata, and edge travel time is length divided by a default speed of 40 km/h, which can be replaced by estimated road speeds (Van Etten, 2020). The graph is stored with NetworkX (Hagberg et al., 2008). Each edge e is then labelled blocked or open. Let φe be the fraction of pixels of e (dilated by 5 pixels) covered by the flooded-road mask, and ℓe the length of its longest uninterrupted flooded stretch. An edge is blocked if $\varphi e \geq \tau$ or $\ell e \geq \ell min$. The second condition reflects that a short submerged stretch is enough to stop a vehicle even when most of the edge is dry. The thresholds $\tau = 0.30$ and $\ell min = 15$ m are chosen on the validation set to maximise the F1 score of blocked-edge detection, with a preference for recall because a missed blocked edge is the more dangerous error [21 – 24].

2.7.Flood-aware routing

Removing the blocked edges from the road graph G gives the accessible graph Gacc. Origin and destination coordinates are snapped to the nearest graph nodes, and the fastest route is found with Dijkstra's algorithm (Dijkstra, 1959) using travel time as the edge weight. If the destination is not in the same connected component as the origin in Gacc, the system reports that no accessible route exists and lists the connected components that are cut off from the origin, which identifies areas that cannot be reached by road. Figure 3 illustrates the effect on a synthetic road network. As shown in Figure 3 Flood-unaware route (A) and flood-aware route (B) between origin

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

O and destination D (illustrative, synthetic network; red: flooded edges).

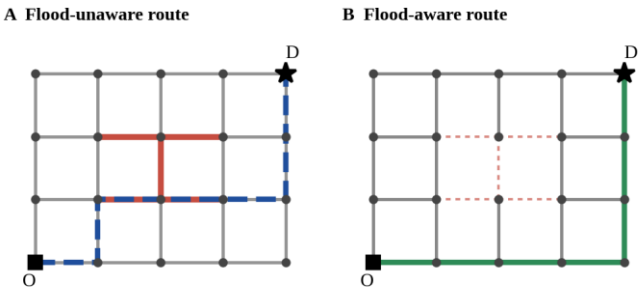


Figure 3 Flood-unaware route (A) and flood-aware route (B) between origin O and destination D (illustrative, synthetic network; red: flooded edges)

2.8.Experimental setup

The 282 labelled tiles are split into training, validation and test sets of about 70%, 15% and 15% (about 197, 42 and 43 tiles). The split is made by geographic block: the tile grid is divided into blocks of 5 × 5 tiles, and whole blocks are assigned to a set with a fixed random seed of 42 (Figure 4). This keeps neighbouring tiles, which look very similar, from appearing in both training and evaluation sets, and so avoids the spatial leakage of a random tile split. The validation set is used for checkpoint selection and for choosing thresholds. The test set is used once, for the final results. As shown in Figure 4 Geographic block split of the tile grid into training, validation and test sets.

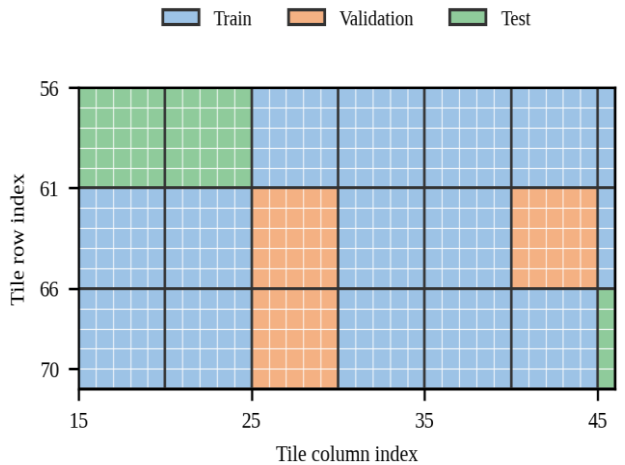


Figure 4 Geographic block split of the tile grid into training, validation and test sets

Training uses the Adam optimiser (Kingma & Ba, 2015) with a learning rate of 1 × 10−4, a batch size of 8 and up to 80 epochs. The learning rate follows cosine annealing (Loshchilov & Hutter, 2017) after a three-epoch linear warm-up, and mixed-precision arithmetic is used. Training stops when the validation flooded-road F1 score has not improved for 15 epochs, and the checkpoint with the highest validation flooded-road F1 is kept. The system is implemented in Python 3.10 with PyTorch (Paszke et al., 2019), segmentation-models-pytorch, Albumentations, rasterio, OpenCV, scikit-image and NetworkX, and runs on a single GPU. As shown in Table 3 lists the settings.

Table 3 Training hyperparameters

Parameter	Value
Input channels	8 (pre + post RGBN)
Crop size	512 × 512
Batch size	8
Maximum epochs	80
Optimiser	Adam
Learning rate	1×10^{-4}
Scheduler	Warm-up + cosine
Loss	Weighted BCE + Dice
Loss weights (λ_f / λ_r)	1.0 / 0.5
Split / seed	70 / 15 / 15 by block, 42
Encoder init.	ImageNet

2.9.Evaluation metrics

Segmentation outputs are scored at pixel level after thresholding the sigmoid output at 0.5, with Dice = 2TP / (2TP + FP + FN), IoU = TP / (TP + FP + FN), precision = TP / (TP + FP), recall = TP / (TP + FN) and F1 = 2PR / (P + R), where TP, FP and FN are true positives, false positives and false negatives. A constant of 10−7 is added to the denominators. Dice and F1 are identical for binary masks, so only F1 is reported. Routing is evaluated on a reference graph built from the CSV geometry of the test mosaics, in which each segment carries its true flooded label. Four route-level quantities are reported for 1,000 randomly sampled origin–destination pairs at least 300 m apart. Route validity is the share of returned routes that contain no reference-flooded segment. Route found rate is the share of pairs for which the system returns a route, among pairs for which a flood-

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

free route exists in the reference graph. Detour ratio is the travel time of the returned route, evaluated on the reference graph, divided by that of the fastest reference flood-free route. Blocked-edge F1 is the F1 score of the blocked or open labelling of edges, where predicted edges are matched to reference segments by overlap within a 5-pixel buffer. A flood-unaware baseline, which routes on the predicted road graph without removing any edge, and the reference graph itself are included for comparison.

3. Results And Discussion

3.1.Segmentation performance

Table 4 gives the pixel-level results of the final model on the held-out test set. The road output reaches an F1 score of 0.79 and the flooded-road output an F1 score of 0.74. For flooded road, precision (0.76) is slightly above recall (0.72), which means the model is more likely to miss a short submerged stretch than to invent one. At tile level, tiles that contain flooded road are detected with a recall of 0.91 and a precision of 0.88. As shown in Table 4. Pixel-level results on the test set.

Table 4 Pixel-level results on the test set

Output	IoU	Precision	Recall	F1
Road	0.65	0.81	0.77	0.79
Flooded road	0.59	0.76	0.72	0.74

3.2.Ablation study

Table 5 isolates the contribution of each design choice. Variant A is the baseline: a post-event RGB model with separate flood and road channels, plain binary cross-entropy and a post-hoc intersection of the two maps. It reaches an F1 score of only 0.42 because its recall is low (0.31): with equal-weight cross-entropy the network is rewarded for predicting background, and the intersection removes every pixel missed by either channel. Adding the Dice term and the positive-class weight and predicting flooded road directly (variant B) raises recall to 0.58 and F1 to 0.63. Adding the near-infrared band (variant C), which is informative for water, brings F1 to 0.68. Adding the pre-event image (variant D), which lets the network use change between the two dates as a cue, raises recall to 0.72 and F1 to 0.74. The loss and output design therefore contributes the largest part of the improvement, and the additional inputs add a further 0.11. As shown in Table 5 Ablation of input, loss and output design (flooded road, test set).

Table 5 Ablation of input, loss and output design (flooded road, test set)

Variant / input	Loss	Output	Recall	F1
A: Post RGB	BCE	AND	0.31	0.42
B: Post RGB	BCE + Dice	Direct	0.58	0.63
C: Post RGBN	BCE + Dice	Direct	0.65	0.68
D: Pre + post RGBN	BCE + Dice	Direct	0.72	0.74

a BCE: binary cross-entropy; AND: intersection of separate flood and road maps; Direct: flooded-road output predicted by the network; RGBN: red, green, blue and near-infrared bands. Variant D is the proposed model.

3.3.Routing performance

Table 6 reports route-level results. Routing on the predicted road graph without removing flooded edges returns a route for almost every pair, but only 64% of these routes are free of flooded segments, so more than a third would lead responders into water. With flooded edges removed, 93% of the returned routes are valid, and a route is found for 95% of the pairs that have a flood-free route. The mean detour is 9% in travel time compared with the fastest reference route. Blocked-edge detection reaches an F1 score of 0.83 (precision 0.81, recall 0.85). The whole pipeline, from a 1300 × 1300 tile to graph and routes, takes about one second per tile on a single GPU. As shown in Table 6. Route-level results on 1,000 origin–destination pairs.

Table 6 Route-level results on 1,000 origin–destination pairs

Method	Validity	Found	Detour
Flood-unaware	0.64	0.99	0.93
Proposed	0.93	0.95	1.09
Reference (labels)	1.00	1.00	1.00

a the detour ratio below 1.00 for the flood-unaware baseline arises because its routes cross flooded

International Research Journal on Advanced Engineering Hub (IRJAEH)
e ISSN: 2584-2137
Vol. 04 Issue: 10 October 2026
Page No: 5708 -5715
https://irjaeh.com
https://doi.org/10.47392/IRJAEH.2026.0741

segments and are therefore shorter than valid routes.

3.4.Discussion

The results show that treating flooded-road detection as a routing problem rather than only a segmentation problem changes what the output is worth. A pixel-level F1 score of 0.74 sounds moderate, but a route validity of 0.93 is much higher, because a route fails only when a missed segment lies on the chosen path, and detours around a blocked edge exist in most urban networks. The remaining invalid routes are caused by missed detections, which is the error that matters most for safety. The thresholds τ and ℓmin were therefore chosen to favour blocked-edge recall over precision, at the cost of occasional unnecessary detours. Each design choice addresses a specific failure. Class imbalance is the main obstacle for road and flooded-road segmentation: with plain cross-entropy the network learns to predict background and recall collapses, as variant A shows, and the weighted loss with a Dice term corrects this. Direct prediction of flooded road avoids the compounding of two independent errors. The near-infrared band and the pre-event image give physically meaningful cues: water absorbs near-infrared light strongly, and the change between dates separates standing water from permanently dark surfaces such as asphalt or shadow. The geographic split makes the reported numbers harder to reach than with a random tile split, because the test tiles come from areas the network has not seen. The metrics are therefore a more honest estimate of performance on new areas, although still for one region.

3.5.Limitations and future work

The study has limits. The dataset is small (282 tiles) and the test set holds about 43 tiles, so confidence intervals are wide. Only one region, Germany, is covered, and generalisation to other road layouts, such as dense Indian cities, and to other sensors is untested. Optical imagery cannot see through cloud or tree canopy, and a mask cannot give water depth, so a segment marked as open is not guaranteed to be passable for every vehicle. Travel times use an assumed default speed. Future work will (i) add synthetic aperture radar imagery, which works through cloud; (ii) estimate water depth classes so that routes can be planned per vehicle type; (iii) replace early fusion with a Siamese encoder; (iv) evaluate on additional regions and on flood events in Indian cities; and (v) wrap the pipeline in a web interface in which responders select an origin and destination and receive the route.

Conclusion

This paper presented a pipeline that identifies accessible routes during floods from satellite imagery. A U-Net with a ResNet34 encoder uses pre-event and post-event four-band imagery to predict roads and flooded roads, a combined weighted cross-entropy and Dice loss counters class imbalance, and a graph-based stage removes flooded segments and returns the fastest remaining route. On a geographically separate test set the model reaches a flooded-road F1 score of 0.74, and 93% of the routes it returns avoid every truly flooded segment, with a mean detour of 9%. The study shows that the step from flooded-road detection to route recommendation is practical and adds value that segmentation alone does not. Larger and more diverse data, radar imagery and depth estimation are the main steps towards operational use.

Acknowledgements

The authors thank [guide / institution name] for guidance and computing support, and the SpaceNet partners for making the SpaceNet 8 dataset publicly available.

References

[1]. Long, J., Shelhamer, E., & Darrell, T. (2015). Fully convolutional networks for semantic segmentation. Proceedings of the IEEE CVPR, 3431–3440.
[2]. Zhang, Z., Liu, Q., & Wang, Y. (2018). Road extraction by deep residual U-Net. IEEE Geoscience and Remote Sensing Letters, 15(5), 749–753.
[3]. Bonafilia, D., Tellman, B., Anderson, T., & Issenberg, E. (2020). Sen1Floods11: A georeferenced dataset to train and test deep learning flood algorithms for Sentinel-1. Proceedings of the IEEE/CVF CVPR Workshops.
[4]. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-Net: Convolutional networks for biomedical image segmentation. MICCAI, LNCS 9351, 234–241.
[5]. He, K., Zhang, X., Ren, S., & Sun, J. (2016).

Deep residual learning for image recognition. Proceedings of the IEEE CVPR, 770–778.

[6]. Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. Proceedings of the IEEE CVPR, 248–255.

[7]. Van Etten, A., Lindenbaum, D., & Bacastow, T. M. (2018). SpaceNet: A remote sensing dataset and challenge series. arXiv:1807.01232.

[8]. Van Etten, A. (2020). City-scale road extraction from satellite imagery v2: Road speeds and travel times. IEEE Winter Conference on Applications of Computer Vision (WACV), 1786–1795.

[9]. Rahnemoonfar, M., Chowdhury, T., Sarkar, A., Varshney, D., Yari, M., & Murphy, R. R. (2021). FloodNet: A high resolution aerial imagery dataset for post flood scene understanding. IEEE Access, 9, 89644–89654.

[10]. Gupta, R., Goodman, B., Patel, N., et al. (2019). Creating xBD: A dataset for assessing building damage from satellite imagery. Proceedings of the IEEE/CVF CVPR Workshops.

[11]. Daudt, R. C., Le Saux, B., & Boulch, A. (2018). Fully convolutional Siamese networks for change detection. Proceedings of the IEEE ICIP, 4063–4067.

[12]. Hänsch, R., Arndt, J., Lunga, D., et al. (2022). SpaceNet 8 – The detection of flooded roads and buildings. Proceedings of the IEEE/CVF CVPR Workshops.

[13]. Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: Fast and flexible image augmentations. Information, 11(2), 125.

[14]. Iakubovskii, P. (2019). Segmentation Models Pytorch. GitHub repository, https://github.com/qubvel/segmentation_models.pytorch.

[15]. Milletari, F., Navab, N., & Ahmadi, S.-A. (2016). V-Net: Fully convolutional neural networks for volumetric medical image segmentation. Proceedings of the International Conference on 3D Vision, 565–571.

[16]. [16]. Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. Proceedings of the IEEE ICCV, 2980–2988.

[17]. Zhang, T. Y., & Suen, C. Y. (1984). A fast parallel algorithm for thinning digital patterns. Communications of the ACM, 27(3), 236–239.

[18]. [18]. van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., et al. (2014). scikit-image: Image processing in Python. PeerJ, 2, e453.

[19]. Douglas, D. H., & Peucker, T. K. (1973). Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. The Canadian Cartographer, 10(2), 112–122.

[20]. Hagberg, A. A., Schult, D. A., & Swart, P. J. (2008). Exploring network structure, dynamics, and function using NetworkX. Proceedings of the 7th Python in Science Conference, 11–15.

[21]. [21]. Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. Numerische Mathematik, 1(1), 269–271.

[22]. Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. International Conference on Learning Representations (ICLR).

[23]. Loshchilov, I., & Hutter, F. (2017). SGDR: Stochastic gradient descent with warm restarts. ICLR.

[24]. Paszke, A., Gross, S., Massa, F., et al. (2019). PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems, 32, 8024–8035.