Vol. 1, Issue 11 | Nov 2026 | Open Access — Research Paper

A HYBRID 3D CNN–AUTOENCODER FRAMEWORK FOR GAS LEAK DETECTION AND SEGMENTATION IN HYPERSPECTRAL IMAGES

G. Vamshi Krishna, G. Kotesh, G. Sai Parthiva, U. Saritha (Guide)
Department of CSE(AI&ML), CMR Technical Campus, Hyderabad, Telangana, India- 501401

***Abstract*—** A gas leak that goes unnoticed can end in a fire, an explosion or a long-running emission, and point sensors only report what happens at the spot where they are bolted down. Hyperspectral cameras record a full spectrum at every pixel and so can show where a gas is, not only that it is present, but the data are bulky and awkward for standard two-dimensional networks. This paper describes a network that works on small hyperspectral patches and does three jobs from one shared code: it reconstructs its input, it decides whether the patch contains a leak, and it produces a mask of the affected pixels. A three-stage 3D convolutional encoder compresses each 16-band, 32×32 patch into a 128-dimensional latent vector. A decoder, a classifier and a segmentation head all read that vector. Training sums the three losses and gives positives a weight of 2.15 because only about a third of the patches contain a leak. Decision thresholds are chosen on the validation split and then frozen. On 462 held-out patches derived from the STARCOP data the model reaches 69.48% accuracy, 50.23% precision, 77.46% recall and a 60.94% F1-score for classification, and a segmentation F1-score of 22.76% (IoU 12.84%). The model finds most leaks but raises many false alarms, and its masks are still coarse. A FastAPI service and a React front end make the trained checkpoint usable from a browser.

***Index Terms*—**Hyperspectral Imaging, 3D CNN, Autoencoder, Gas Leak Detection, Semantic Segmentation, Deep Learning, Spatial-Spectral Features

I. INTRODUCTION

Refineries, chemical plants, pipelines and storage yards all handle gases that are flammable, toxic or both. When such a gas escapes, the first hours matter most, and the people responsible for a site would like to be told early and told where. Fixed detectors do give an alarm, but each one samples a single location, so covering a large facility takes many of them, and a plume that drifts between two units may never reach either.

Imaging spectrometers approach the problem differently. Instead of three colour values, every pixel carries a reflectance curve sampled in many narrow bands, and a gas leaves a faint absorption pattern in some of those bands [1], [2]. The same property that makes the data informative makes it hard to use. A scene is a cube with two spatial axes and one spectral axis, neighbouring bands are strongly correlated, and the signal of interest can be weaker than the variation between ground surfaces. A 2D network that stacks the bands as channels mixes them with its first layer and has no explicit notion of spectral order.

A 3D convolution slides its kernel along the spectral axis as well as across the image, so a single filter can respond to a pattern that spans a few adjacent bands and a small neighbourhood of pixels [7], [8]. We build on this idea and add an autoencoder objective [9]. The reasoning is that a latent code forced to reconstruct the whole patch has to keep information about the background and the spectral structure, and that this should give the classifier and the segmentation head a better starting point than a code learned from the labels alone.

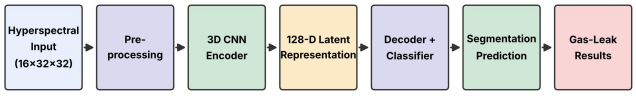


Fig. 1. High-level flow of the proposed gas-leak detection pipeline.

The objectives of the work are to detect leaks in hyperspectral patches automatically, to learn a compact spatial-spectral representation, to classify each patch as leak or no leak while also localising the gas pixels, to measure all of this on data that played no part in training or tuning, and to put the trained model behind a simple web interface. The main contributions are:

1) A 3D CNN encoder whose 128-dimensional output is shared by a reconstruction decoder, a leak classifier and a segmentation head, so that all three tasks train one representation.

2) A weighted joint loss in which the positive class is up-weighted according to the class ratio in the training split, together with classification and segmentation thresholds that are selected on validation data only.

3) A test report on 462 unseen patches that includes the confusion matrix and the segmentation overlap, including the numbers that are unfavourable to the model.

4) A working FastAPI and React application built around the trained checkpoint.

Section II reviews related work. Section III describes the data, Section IV the model and its training, and Section V the software. Results are discussed in Section VI and Section VII concludes.

© 2026 IJVAST | ISSN: 2839-5379 | DOI: 10.68296/IJVAST20260096 | Open Access — CC BY 4.0

II. RELATED WORK

A. Gas Detection with Sensors and Imaging

Catalytic, electrochemical and infrared point sensors are the standard equipment for leak monitoring. They are cheap and sensitive, but each reading belongs to one place, and the layout of the sensors decides what can be seen. Remote sensing removes that constraint. Methane in particular has been studied from aircraft and satellites because it absorbs in the shortwave infrared, and such observations are now used to estimate emissions at facility and regional scale [2]. Airborne imaging spectrometers can also locate individual plumes in near real time [3].

B. Classical Hyperspectral Detectors

The Spectral Angle Mapper compares the angle between a pixel spectrum and a reference spectrum [4], and the Adaptive Cosine Estimator tests a pixel against a target signature using the background covariance [5]. Both work when the target signature is known and the background is reasonably well modelled. Their behaviour degrades when the signature is mixed with a varied background, and generic classifiers such as SVMs and random forests need hand-picked spectral features to do better.

C. Deep Networks for Hyperspectral Data

Chen et al. [6] showed that convolutional networks learn useful spectral-spatial features from hyperspectral images without manual feature design. Li et al. [7] applied 3D convolutions directly to image cubes for classification, an idea that goes back to the use of 3D kernels for video [8]. For methane specifically, Ruzicka et al. [1] released the STARCOP benchmark and trained segmentation networks on AVIRIS-NG and WorldView-3 imagery, which is the source of the patches used here.

D. Autoencoders

An autoencoder squeezes its input through a narrow code and is trained to rebuild it [9]. Because no labels are required, the reconstruction term can be added to a supervised objective as a regulariser. The code in our model is read by three heads at once, which ties the supervised tasks to a representation that must also describe the raw spectra.

E. Open Problems

Taken together, the literature leaves a few problems that shape this work: point sensors cover little area; signature-based detectors rely on a spectrum chosen in advance; the spectral and spatial structure of the cube is not exploited together by 2D models; hyperspectral data has many bands and few labelled samples; positives are rare; and a patch-level yes/no answer does not say where in the patch the gas is. We address these in the sections that follow.

III. DATASET

A. Source and Split

Samples are patches cut from the STARCOP data [1]. Each patch keeps 16 spectral bands over a 32×32 pixel window, and carries a binary label (gas leak or no gas leak) together with a pixel mask for the gas region. The 3,263 patches are divided into training, validation and test subsets as listed in Table I. Validation data was used to monitor training and to pick thresholds; the test subset was scored once with those settings fixed.

TABLE I
PATCHES PER SPLIT

Split	Total	No leak	Leak
Training	2,343	1,600	743
Validation	458	316	142
Testing	462	320	142
Total	3,263	2,236	1,027

B. Class Imbalance

Leaks are the minority class. Positives make up 31.7% of the training patches (743 of 2,343), and the validation and test subsets contain 31.0% and 30.7%. A network that always answered “no leak” would therefore be right roughly 68 to 69% of the time while detecting nothing, which is worth remembering when reading the accuracy figures in Section VI. The ratio of negatives to positives in the training split, 1,600 to 743, is about 2.15, and this is the weight given to positives in the loss.

C. Input Representation

Each patch is arranged as a single-channel volume so that a 3D convolution can treat the 16 bands as a depth axis. The tensor shape, in the order channel, bands, height, width, is

$$input = 1 \times 16 \times 32 \times 32 \quad (1)$$

IV. PROPOSED METHODOLOGY

A. Overview

The architecture is drawn in Fig. 2. A patch enters the encoder, which produces a 128-dimensional code z. Three heads consume z: a decoder that rebuilds the 16-band patch, a classifier that outputs a leak probability, and a segmentation head that outputs a gas-region mask. Only the encoder is shared, which makes it the place where the three objectives meet.

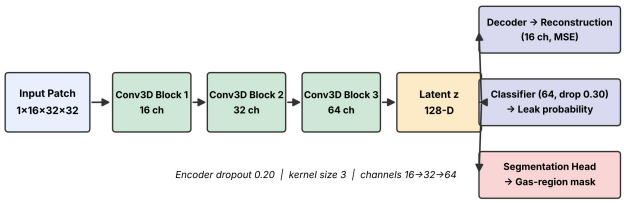


Fig. 2. Architecture of the Hybrid 3D CNN–Autoencoder. One encoder produces the shared latent code z, which feeds the decoder, the classifier and the segmentation head.

B. Preprocessing

Patches are loaded and reshaped into the 1×16×32×32 layout, their values are normalised, and the labels and segmentation masks are prepared as training targets. The split into training, validation and test subsets is fixed before any model is trained, so that no patch appears in more than one subset.

C. 3D CNN Encoder

The encoder has three convolutional stages whose channel counts grow from 16 to 32 to 64. All kernels have size 3, so each filter spans three neighbouring bands and a small pixel window, and dropout of 0.20 is applied for regularisation. Because the kernel moves along the band axis, the first stage already sees relationships between neighbouring wavelengths instead of treating them as independent channels. Later stages combine these into patterns that span larger parts of the spectrum and of the patch.

D. Latent Code

The output of the last stage is projected to a 128-dimensional vector. This is a large reduction from the 16,384 values of the input, which is deliberate: the code has to be small enough that the decoder cannot simply copy the input, and rich enough that the two other heads can use it.

E. Decoder

The decoder maps z back to a 16-band patch. Its loss is the mean squared error between the output and the input. The reconstruction term does not need labels, and its job is to stop the encoder from discarding information that is irrelevant to the leak label but present in the spectra.

F. Classification Branch

The classifier is a small network with a hidden layer of 64 units, dropout of 0.30 and two output classes, and it reports the probability that the patch contains a leak. It is trained with binary cross-entropy on logits, with the positive class weighted as described below.

G. Segmentation Branch

A patch-level answer says that gas is present but not where. The segmentation head therefore predicts a per-pixel mask of the gas region, which is compared with the annotated mask. Overlap is measured with precision, recall, F1-score and intersection over union (IoU).

H. Joint Loss

The three objectives are added with fixed weights:

$$L_{total} = \lambda_{rec} L_{rec} + \lambda_{cls} L_{cls} + \lambda_{seg} L_{seg} \quad (2)$$

with $\lambda_{rec} = 0.5$, $\lambda_{cls} = 1.0$ and $\lambda_{seg} = 1.0$. Reconstruction is given the smallest weight, so that it acts as a regulariser and not as the main goal. The classification and segmentation losses both use a positive-class weight of 2.15, so a missed positive costs about twice as much as a false alarm during training.

I. Training Settings

Table II lists the configuration of the model and optimiser. We use AdamW [11], which applies weight decay separately from the gradient update.

TABLE II
MODEL AND TRAINING CONFIGURATION

Parameter	Value
Input (channels / bands)	1 / 16
Patch size	32 × 32
Encoder channels	16, 32, 64
Kernel size	3
Encoder dropout	0.20
Latent dimension	128
Classifier hidden units / dropout	64 / 0.30
Output classes	2
Decoder output channels	16
Loss weights (rec / cls / seg)	0.5 / 1.0 / 1.0
Positive-class weight	2.15
Optimiser	AdamW
Learning rate	0.001
Weight decay	0.0001

J. Threshold Selection

Probabilities are turned into decisions with two thresholds, one for the patch label and one for the segmentation mask. Rather than assume 0.5, we try candidate values on the validation predictions and keep the best. For the checkpoint evaluated here this gave 0.51 for classification and 0.23 for segmentation. The values were then applied to the test subset unchanged, so the test patches had no influence on them. That the segmentation threshold is so low suggests the mask probabilities are underconfident, a point we return to in Section VI.

V. IMPLEMENTATION

The model is written in Python with PyTorch [12], and hyperparameters are read from a YAML configuration file. Training and evaluation produce a PyTorch checkpoint, and the same file is loaded for inference. The web application has two parts. The backend is a FastAPI [13] service, whose entry point is backend/app/main.py, run with Uvicorn; it loads the checkpoint and exposes the model to the interface. The frontend is a React application bundled with Vite, kept in the frontend directory. Development was done in Visual Studio Code.

To run the system locally, the backend is launched from the project root on 127.0.0.1, port 8000, and the frontend is started in a second terminal with npm run dev. Both processes have to stay up while the application is used, and the page is opened at the address that Vite prints.

VI. RESULTS AND DISCUSSION

A. Metrics

Classification is scored with accuracy, precision, recall and the F1-score. With TP, TN, FP and FN denoting true positives, true negatives, false positives and false negatives:

$$Accuracy = (TP + TN) / (TP + TN + FP + FN) \quad (3)$$

$$Precision = TP / (TP + FP), \; Recall = TP / (TP + FN) \quad (4)$$

$$F1 = 2 \cdot Precision \cdot Recall / (Precision + Recall) \quad (5)$$

Segmentation is scored with the F1-score and IoU over pixels. For a single mask the two are linked by IoU = F1 / (2

− F1), which is a convenient consistency check on the reported values.

B. Classification Results

The checkpoint was evaluated on the 462 test patches with the thresholds of Section IV-J. Table III gives the classification results together with the individual loss terms. The total of 1.2408 is the weighted sum 0.5 × 0.0208 + 0.7438 + 0.4866, as expected from the loss definition.

TABLE III
CLASSIFICATION AND LOSS RESULTS ON THE TEST SET

Metric	Test result
Accuracy	69.48%
Precision	50.23%
Recall	77.46%
F1-score	60.94%
Total loss	1.240837
Reconstruction loss	0.020779
Classification loss	0.743813
Segmentation loss	0.486634

C. Confusion Matrix

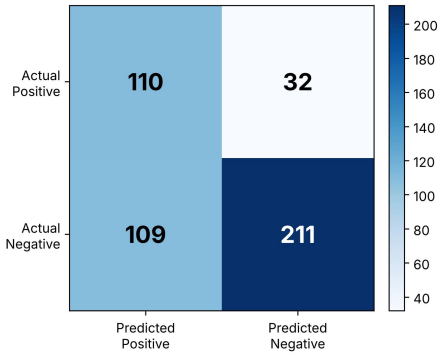


Fig. 3. Confusion matrix on the 462-patch test set.

The matrix in Fig. 3 shows 110 leaks found, 32 leaks missed, 211 clean patches correctly cleared and 109 clean patches wrongly flagged. Recall is therefore 110/142 = 77.46%, while precision is only 110/219 = 50.23%. Other figures that follow from the same counts are a specificity of 65.9%, a balanced accuracy of 71.7% and a negative predictive value of 86.8%. The model labels 219 of the 462 patches as leaks although only 142 are, so about half of the alarms are false.

D. Segmentation Results

TABLE IV
SEGMENTATION RESULTS ON THE TEST SET

Metric	Test result
Precision	16.68%
Recall	35.83%
F1-score	22.76%
IoU	12.84%

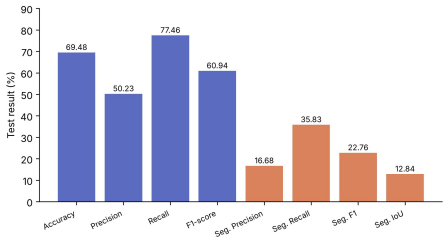


Fig. 4. Summary of classification and segmentation test metrics.

The masks overlap the annotation only weakly (Table IV). Recall is about twice precision, so the predicted regions tend to be larger than the true ones or to include pixels that are not gas. The low segmentation threshold of 0.23 pushes in the same direction: it was chosen to recover more gas pixels, at the price of including many wrong ones.

E. Discussion

Classification. The shared code carries enough spatial-spectral information to find most leaks. A recall of 77.46% against a positive rate of 30.7% is clearly better than chance, and the weighting of positives during training and the validation-based threshold both point toward this behaviour. Accuracy alone (69.48%) is a poor summary here, since it is close to the 69.3% that a classifier labelling everything negative would score on this test set.

False alarms. The cost of the recall is precision. With 109 false positives against 110 true ones, a person reading the alerts would find about half of them unfounded. In a monitoring setting that would quickly erode trust, so reducing false positives is the first thing to work on. Stronger regularisation, more negative examples that resemble leaks, or a higher decision threshold are all options.

Segmentation. An F1-score of 22.76% and an IoU of 12.84% mean that the localisation is not yet usable on its own. The segmentation branch reads the 128-dimensional code, which is much smaller than the patch, so fine spatial detail has to be recovered from a very compressed description. Whether skip connections from the encoder, as in U-Net [10], would help is untested here.

Overall. The results show that the approach works well enough to be worth developing, but they do not show that it is reliable enough for deployment at an industrial site.

F. Advantages and Limitations

- Spectral and spatial structure are learned together in one compact code that serves three tasks.
- Class imbalance is handled in the loss, and thresholds are chosen without using test data.
- The trained checkpoint is wrapped in a web application, so it can be tried without writing code.
- Precision of 50.23% means false alarms are common.
- Segmentation quality is low (IoU 12.84%) and is not suitable for locating plumes in practice.
- Only one test split from one data source was used; behaviour under other sensors, sites and weather conditions has not been measured.

- Real-time operation has not been assessed.

VII. CONCLUSION AND FUTURE WORK

We presented a hybrid 3D CNN and autoencoder for hyperspectral gas-leak analysis in which one encoder feeds a decoder, a classifier and a segmentation head. On 462 held-out STARCOP-derived patches it finds 77.46% of the leaks and reaches 69.48% accuracy, but precision is 50.23% and segmentation overlap is low, with an F1-score of 22.76% and an IoU of 12.84%. The work therefore shows that the architecture can be trained and deployed, and it also shows clearly where it falls short.

Future work will concentrate on reducing false positives and improving the masks, for example by adding skip connections or attention to the segmentation path. We also plan to evaluate the model on other hyperspectral datasets, to speed up inference on GPUs and edge hardware, and to improve how predicted gas regions are drawn in the interface.

REFERENCES

[1] V. Ruzicka, G. Mateo-Garcia, L. Gómez-Chova, A. Vaughan, L. Guanter, and A. Markham, "Semantic segmentation of methane plumes with hyperspectral machine learning models," *Scientific Reports*, vol. 13, Art. no. 19999, 2023.

[2] D. J. Jacob et al., "Satellite observations of atmospheric methane and their value for quantifying methane emissions," *Atmospheric Chemistry and Physics*, vol. 16, no. 22, pp. 14371–14396, 2016.

[3] D. R. Thompson et al., "Real-time remote detection and measurement for airborne imaging spectroscopy: a case study with methane," *Atmospheric Measurement Techniques*, vol. 8, no. 10, pp. 4383–4397, 2015.

[4] F. A. Kruse et al., "The spectral image processing system (SIPS)—interactive visualization and analysis of imaging spectrometer data," *Remote Sensing of Environment*, vol. 44, no. 2–3, pp. 145–163, 1993.

[5] S. Kraut, L. L. Scharf, and L. T. McWhorter, "Adaptive subspace detectors," *IEEE Transactions on Signal Processing*, vol. 49, no. 1, pp. 1–16, 2001.

[6] Y. Chen, H. Jiang, C. Li, X. Jia, and P. Ghamisi, "Deep feature extraction and classification of hyperspectral images based on convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 54, no. 10, pp. 6232–6251, 2016.

[7] Y. Li, H. Zhang, and Q. Shen, "Spectral–spatial classification of hyperspectral imagery with 3D convolutional neural network," *Remote Sensing*, vol. 9, no. 1, Art. no. 67, 2017.

[8] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[9] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[10] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

[11] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. International Conference on Learning Representations (ICLR)*, 2019.

[12] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, vol. 32, 2019, pp. 8024–8035.

[13] FastAPI documentation. [Online]. Available: https://fastapi.tiangolo.com

© IJVAST — International Journal of Vast Advancements in Science and Technology.
ISSN: 2839-5379 | DOI: 10.68296/IJVAST20260096 | Open Access — CC BY 4.0 License