AutoFocal Loss with Lightweight DNN for Stroke Prediction in Fog Computing

Vijayetha Thoday
Department of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai, India
vijayethap@gmail.com (corresponding author)

Mary Posonia
Department of Computer Science and Engineering, Sathyabama Institute of Science and Technology, Chennai, India
maryposonia.cse@sathyabama.ac.in

Received: 9 January 2026 | Revised: 8 February 2026 and 18 February 2026 | Accepted: 19 February 2026

 DOI: https://doi.org/10.48084/etasr.17446

ABSTRACT

The growing volume of Internet of Things (IoT) healthcare data makes efficient patient monitoring and disease prediction increasingly important. Cloud Computing (CC) enables scalable storage and processing, but struggles with latency-sensitive applications. Fog Computing (FC) addresses this issue by placing resources closer to users. Existing fog-based Deep Learning (DL) models for disease monitoring often perform poorly on imbalanced healthcare datasets, are biased toward the majority class, and are typically designed for large cloud systems. This paper introduces a fog-based framework for real-time stroke prediction using a lightweight, quantization-optimized DNN with automatically weighted focal loss. The proposed model handles severe class imbalance, maintains diagnostic accuracy, and reduces model size for fog deployment. Experiments on a public stroke prediction dataset show that the proposed system achieves higher AUPRC, recall, and F1-score than baseline models, while remaining compact enough for fog computing.

Keywords-internet of things; healthcare fog computing; Deep Neural Network (DNN); weighted focal loss; post training quantization

I. INTRODUCTION

The rapid growth of the Internet of Things (IoT) has led to massive data generation across domains like smart cities, homes, waste management, and healthcare [1]. In healthcare, IoT technologies are used for better patient care, remote monitoring, and to improve healthcare outcomes [2-4]. Cloud Computing (CC) is a widely used technology that builds these interconnected IoT device networks with powerful computation and massive storage. After collecting patient data through wearable sensors or medical devices in hospital wards, the data are analyzed and stored on the cloud. The timely results are then shared with family, healthcare providers, and doctors for further action. Scalability and multitenancy are some of the features of the cloud, making it affordable for the end user. However, healthcare applications are latency-sensitive, and delays in monitoring or alerts are unacceptable [5-7]. Cloud servers sit far from users, introducing latency that these systems cannot tolerate. Hence, Fog Computing (FC) has emerged as a middle-layer architecture. Its proximity to end devices cuts latency and boosts throughput compared to the cloud. FC nodes are near the user. Due to this geographical proximity of the fog nodes to the end devices, latency, throughput, and response time improve in the fog compared to the cloud. Consequently, sensor-fog-cloud systems constitute a new paradigm for smart healthcare where the intelligence and processing power are held in the local gateways, routers, and servers. They are nodes with limited resources for computation and storage. The nodes can be deployed in locations like hospitals, cafes, restaurants, and traffic signals. For persistent long-term storage and specialized computation, the data are moved to the cloud. The characteristics and challenges of the FC are mobility of the fog nodes, heterogeneity, low latency, and location awareness [8-10].

Fog healthcare research is categorized into three main areas: frameworks for Healthcare 4.0, DL/metaheuristic systems for prediction, and data security [11-13]. Authors in [14-20] covered ICU monitoring, cardiac arrhythmia, stress, diabetes, dengue, dementia, elderly care, and tumor detection, all using a three-layer fog architecture that outperforms cloud-based systems for real-time vital sign monitoring. Fog nodes even enable geospatial tracking for important alerts.

However, a significant challenge remains: deploying DL models on fog devices is hard, since these devices have limited computational and memory resources [21].

The healthcare condition called "Stroke", caused by an abrupt interruption of blood flow to the brain, is considered the second leading cause of death worldwide. This medical emergency is marked by various factors such as age, habits, and stress, which can be a life-threatening condition. Early detection is crucial, and hence various machine learning models have been developed for stroke prediction. These models can aid in early diagnosis, thus helping the patient and healthcare providers. Traditional machine learning approaches [22] for stroke prediction achieve strong accuracy; however, real-time deployment on hospital servers or resource-constrained edge/fog devices is a challenge.

Therefore, the research contributions of the proposed work are:

- A fog-based integrated solution for stroke risk prediction on fog healthcare devices.
- Automatically Weighted Focal Loss is used due to the nature of the imbalanced dataset. Unlike standard focal loss with fixed weights, the proposed approach dynamically updates class weights during training based on per-class learning performance.
- The proposed neural network is optimized for fog device deployment using post-training quantization, achieving 4x model compression.

A quick look at prior DNN-based fog healthcare models follows:

- FETCH – FogBus platform runs DL on fog workers [23]. Good system orchestration, but no lightweight architecture for edge constraints.
- HealthFog – Heart disease diagnosis [24]. Lightweight fog service, effective.
- HAWKFOG – Balances accuracy and latency, but networks stay heavy. No parameter minimization [25].
- FRIEND – Ensemble DL for cardiovascular disease [26]. Compares fog versus cloud on latency, jitter, and throughput.
- CNN-based real-time stroke monitoring [27]. Works, but lacks model size and latency data on fog nodes.
- FogDLearner – Cardiac health on PureEdgeSim [28]. Focuses on QoS and accuracy.
- Ten-layer DNN for stroke with SMOTE + Fast ICA [29]. Reports AUC=1.00, but ignores AUROC and SMOTE leakage on imbalanced data.
- Heart disease prediction (DNN, RF, NB) [30]. Hybrid NB+RF with feature selection wins.
- Deep dense DNN with SMOTE + hyperparameter tuning [31]. Reported AUROC=0.94.
- Ensemble neural network (no SMOTE) [32]. Random Forest beats DNN. AUPRC limits of DL on small tabular stroke data.

On model compression: Authors in [33] showed that INT8 quantization shrinks CNN weights/activations, while authors in [34] optimized MobileNetV3Large for edge deployment—accuracy holds, size drops.

The gap: Most fog healthcare papers focus on infrastructure, not the intelligence layer. DNNs give high accuracy but hog bandwidth and latency. The present work fills this gap by building a DNN that boosts minority-class sensitivity, stays lightweight, and actually fits on resource-constrained fog nodes.

II. PROPOSED METHODOLOGY

The Smart Healthcare System (SHS) uses a three-layer hierarchy:

- Sensor Layer: Wearables and hospital monitors collect real-time clinical data: age, gender, hypertension, heart disease history, BMI, smoking status, and more.
- Fog Layer: A distributed set of compute nodes preprocesses raw data (normalization, outlier removal) and transmits 11-dimensional feature vectors. These nodes run a quantized neural network with Automatically Weighted Focal Loss to handle class imbalance and predict stroke risk.
- Cloud Layer: A central server collects prediction logs from all fog nodes, performs patient-level analytics, validates model performance, and periodically re-trains the network on aggregated data. Updated quantized weights are then pushed back to fog nodes for continuous model improvement.

Figure 1 illustrates the fog system architecture.

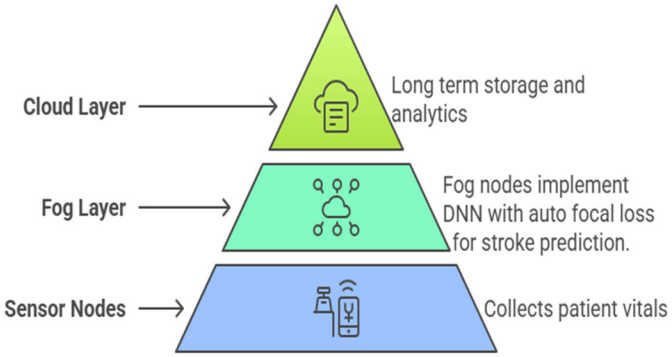

Fig. 1. Fog-stroke healthcare architecture.

A. *Fog Layer*

The complete working pipeline of this layer is developed in 4 steps:

1) *Step-1: Data Preprocessing*

Convert raw sensor data into a standardized feature vector for DNN inference. The dataset has 10 predictive features (mixed continuous and categorical) plus a binary target (0/1), capturing clinical and lifestyle information. The ID column is dropped, and one-hot encoding is applied, yielding a 12-

dimensional vector. Continuous features are standardized with StandardScaler, fitted only on training data to prevent leakage. This ensures stable gradients across feature scales, critical for training on imbalanced health data. Missing values (e.g., BMI) are imputed with the median. The standardized feature vector is given as $\hat{x} \in R^{\mathbb{12}}$.

2) *Step-2: Neural Network Forward Pass*

DNNs excel at learning hierarchical, nonlinear patterns from multivariate patient data. Conventional feed-forward networks stack dense layers; nonlinear activations refine hidden representations into discriminative features. They approximate complex functions, excelling at risk prediction, event forecasting, and state classification [35].

A standard DNN pipeline includes input normalization, hidden-layer transforms, supervised loss (e.g., cross-entropy), and backpropagation updates. However, standard DNNs struggle with highly imbalanced health datasets. They favor high specificity while missing positive cases and recall suffers. Dense networks also use uniform projections, hurting convergence and minority-class sensitivity. Deploying them on resource-constrained fog devices is another hurdle.

These issues are addressed with a tailored DNN (Figure 2). It uses a 12-128-64-32-1 architecture with batch normalization before ReLU on layers 1 and 2, progressive dropout (0.35 → 0.25 → 0.2), and ReLU in every layer. BatchNorm stabilizes gradients when positive examples are rare. Higher dropout rates prevent majority-class overfitting. ReLU adds sparsity and efficient gradient flow, helping the network learn stroke risk patterns directly from patient data. The three hidden layers are given as:

$$h^{(1)} = BN^1\left(ReLU(W^1x + b^1)\right) \odot (1 - D^{0.35}) \in \mathbb{R}^{128}$$

$$h^{(2)} = BN^2\left(ReLU\left(W^2h^{(1)} + b^2\right)\right) \odot (1 - D^{0.25}) \in \mathbb{R}^{64}$$

$$h^{(3)} = ReLU\left(W^3h^{(2)} + b^3\right) \odot (1 - D^{0.2}) \in \mathbb{R}^{32}$$

Sigmoid activation for probability transforms the logit into a stroke probability in [0,1]. It is used for risk classification and focal loss calculation.

$$p = \sigma\left(W^4 h^{(3)} + b^4\right) \in [0,1] \quad (1)$$

3) *Step-3: Automatic Weighted Focal Loss*

Given the highly imbalanced outcome (only a small fraction of positive cases), the proposed model incorporates the focal loss, as presented in [36]. Although it was initially proposed for a multiclass problem, the current work adapted it for stroke prediction as a binary class task. Per class, F1-scores determine dynamic weights given as:

$$w_c^e = \frac{1-\varphi_c^e}{\max\left(1-\varphi_0^e, 1-\varphi_1^e\right)} \quad \text{for } c \in \{0,1\} \quad (2)$$

where φ_0^e and φ_1^e are the F1-scores for the no-stroke and stroke classes at epoch e, respectively. To consider performance from all previous epochs, an Exponential Moving Average (EMA) is employed:

$$w = \alpha w^e + (1 - \alpha)w \quad (3)$$

where $\alpha = 0.5$. The dynamic weighted focal loss for binary class prediction is:

$$L = w_0^e(1 - \hat{y})^\gamma \log(1 - \hat{y}) + w_1^e(\hat{y})^\gamma \log(\hat{y}) \quad (4)$$

where ŷ is the predicted stroke case probability, w_0^e *and* w_1^e are dynamic weights for classes 0 and 1, and $\gamma = 2.0$. This approach ensures that the minority positive stroke class receives higher weights when underperforming, preventing the majority class from dominating the gradient.

4) *Step-4: Calibration and Post Training Quantization*

Temperature scaling with T=0.128 is applied post-hoc to sharpen DNN probability outputs, thereby reducing decision boundary uncertainty. This single-parameter calibration sharpens decision boundaries, lowering the optimal F2 threshold from 0.6 to 0.48 and increasing sensitivity from 50% to 72%.

$$p_{(cal)} = sigmoid(logit(p)/T) \quad (5)$$

To enable the deployment of the trained model on resource-constrained fog devices, post-training quantization is employed to reduce model size while preserving prediction accuracy. This converts the float32 Keras model into compact TFLite formats. Dynamic-range quantization produces lightweight 28KB models, whereas full INT8 quantization, calibrated using representative training data, yields 12KB models with negligible accuracy degradation.

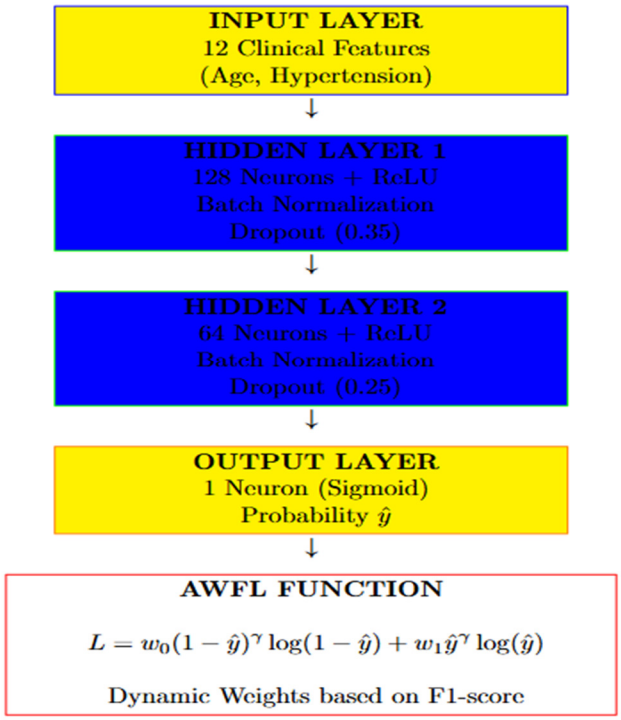

Fig. 2. DNN auto focal loss.

Finally, quantization simulates floating-point arithmetic:

$$q = \text{round}(x / scale + \text{zero\_point}) \in Z_s$$

At inference time, dequantization is performed using learned parameters:

$$\hat{x} = scale(q - \text{zero\_point}) \approx x \quad (6)$$

where x is the original float32 value, q is a quantized 8-bit integer, $scale$ is a positive float32 that scales the range, and zero_point is the integer value that corresponds to the real-world value of 0.

III. EXPERIMENTATION

Experiments were conducted using the stroke prediction dataset [37], which contains 5,110 patient records. Among these, 4,861 patients are without stroke, and 249 patients have experienced a stroke, resulting in an approximate 20:1 class imbalance (positive stroke cases comprise only 4.8% of the dataset). The prediction task is a binary classification.

Stratified 5-fold cross-validation was employed to ensure generalization by preserving the original class distribution in each fold. In each fold, 80% of the data were used for model development and split into 61% for training and 19% for validation using stratified sampling. The remaining 20% were held out as the test fold, which was never seen during training or hyperparameter tuning.

The model was trained for 150 epochs with mini-batches of size 128 and an Adam optimizer with learning rate η = 0.002. The proposed automatic focal loss uses γ = 2.5, with the α parameter updated via EMA, β = 0.9 for each epoch. ReLU activations, progressive dropout, and batch normalization were employed after layers 1 and 2.

All experiments were implemented in Python/TensorFlow on a system with an Intel Core i7 processor, Windows 11, and 16 GB of RAM. The study considered 15 fog nodes divided into three tiers. Tier-1 simulates microcontrollers with memory <1 MB; Tier-2 simulates devices like Raspberry Pi with a 512 MB memory limit; and Tier-3 has a larger memory limit of 2–5 GB, representing fog devices used as hospital servers. All tiers have CPU speeds ranging from 200 MHz to 2.5 GHz.

- Area Under the Receiver Operating Characteristic Curve (AUROC) is a summary statistic that indicates the overall effectiveness of a classification model. The ROC curve plots the True Positive Rate (TPR) versus the False Positive Rate (FPR) across different threshold levels. TPR measures the fraction of actual positives correctly identified, whereas FPR represents the fraction of negatives incorrectly classified as positives:

$$\text{TPR} = \frac{\text{TP}}{(\text{TP+FN})}$$

$$\text{FPR} = \frac{\text{FP}}{(\text{FP+TN})}$$

$$\text{AUROC} = \int_0^1 \text{TPR}(t)\, d\text{FPR}(t) \quad (7)$$

- Sensitivity (also known as recall or TPR) measures the model's ability to correctly identify positive stroke cases:

$$\text{Sensitivity} = \frac{\text{TP}}{(\text{TP+FN})} \quad (8)$$

- Specificity evaluates the ability to correctly identify low-risk patients:

$$\text{Specificity} = \frac{\text{TN}}{(\text{TN+FP})} \quad (9)$$

- Precision measures the reliability of positive predictions:

$$\text{Precision} = \frac{\text{TP}}{(\text{TP+FP})} \quad (10)$$

- F1-score merges precision and recall into a unified metric:

$$\text{F1} = 2 \cdot \frac{\text{Precision} \cdot \text{Sensitivity}}{\text{Precision+Sensitivity}}$$

Since the optimal clinical threshold is not known a priori, the decision threshold that maximizes the F1-score over the validation set is computed as:

$$t^* = \arg\max_t \text{F1}(t) \quad (11)$$

In addition to predictive performance, fog suitability was evaluated using deployment-oriented metrics: inference latency and model size.

- Inference Latency: Per-sample prediction time in milliseconds:

$$\text{Inference Latency} = \frac{1}{N}\sum_{i=1}^{N}\left(t_{end(i)} - t_{start(i)}\right) \quad (12)$$

where t_{end} is the timestamp after output generation, t_{start} the timestamp before the forward pass, and N is the number of samples.

- Model Size: Memory footprint after quantization, measured in KB.

IV. RESULTS AND DISCUSSION

The predictive performance of the proposed model is presented in comparison with three baselines: a deep feedforward Artificial Neural Network (ANN) with four hidden layers using ReLU activation and an output layer of a single neuron with sigmoid activation (trained with Adam optimizer); an SVM with an RBF kernel and regularization parameter C=1.0; and a random forest classifier with 100 decision trees, using stratified sampling to preserve class distribution across training and test sets. All models were evaluated using AUROC, AUPRC, and F1-score, with the F1-score computed at the threshold that maximized its value on the validation set. Table I displays the mean ± standard deviation across folds. Although random forest achieved a comparable AUROC value, its AUPRC, F1-score, and recall were lower than those of the proposed model, a consequence of the severe 20:1 class imbalance in the dataset. The proposed model attained the highest AUPRC, F1-score, and recall among all compared models. The relatively low standard deviations across folds demonstrate the stability and consistency of the proposed model in detecting minority-class stroke cases. These results confirm that standard models without explicit imbalance-handling mechanisms underperform on minority-class-sensitive metrics, whereas the proposed AWFL-based DNN maintains a superior precision–recall balance suitable for real-time stroke prediction on fog devices. The confusion matrix of the proposed model is shown in Figure 3.

TABLE I. METRIC RESULTS

Model	AUROC	AUPRC	F1	Recall
Proposed DNN with AWFL	0.81±0.03	0.18±0.03	0.24±0.04	0.47±0.12
ANN	0.71±0.02	0.12±0.01	0.14±0.02	0.17±0.05
SVM	0.61±0.04	0.11±0.01	0.13±0.02	0.16±0.06
Random Forest	0.80±0.03	0.16±0.03	0.23±0.02	0.43±0.14

TABLE II. INFERENCE LATENCY

Fog-tier	Inference latency(ms)
Tier-1	7.5ms
Tier-2	2.8ms
Tier-3	1.4ms

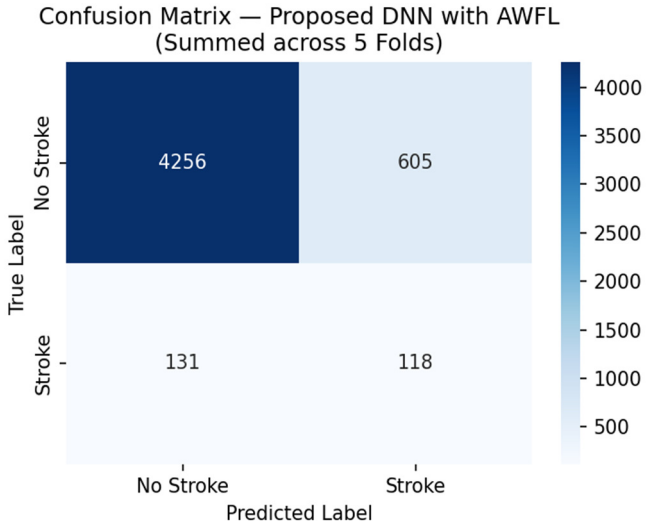

Fig. 3. Confusion matrix.

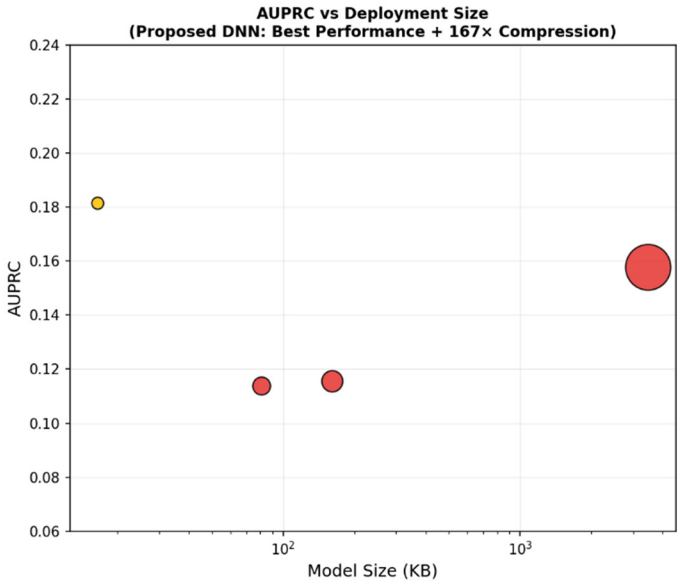

Fig. 4. Model size post quantization.

The study also measured the post-quantization model size. The proposed model achieves an AUPRC of 0.18 at a deployment size of 12 KB, whereas random forest requires 2 MB to achieve a lower AUPRC of 0.16, 167× size reduction with superior minority-class detection performance. As depicted in Figure 4, the yellow marker highlights the best trade-off between predictive performance and deployment size. The model sizes of all baselines confirm that only the proposed model is well-suited for real-time stroke prediction on resource-constrained fog devices across all three tiers.

The measured inference latency across the three fog-tier nodes is illustrated in Table II. The quantized AWFL-based DNN achieves favorable inference latency. Reported values are averaged over 30 inference runs to ensure stability, and no processing overhead is included in the measurements. These results demonstrate that the proposed model satisfies real-time inference requirements even on resource-constrained fog devices, whereas higher-tier nodes further reduce latency due to their enhanced computational capability.

V. CONCLUSION AND FUTURE WORK

Wearable IoT devices generate real-time health data for stroke prediction. Fog Computing (FC) brings processing closer to users, reducing cloud dependence and enabling faster decisions. The DNN works well for stroke risk prediction using vital signs. However, stroke datasets are highly imbalanced, with few positive cases, which biases models and lowers sensitivity for the minority class.

This study introduces an automatic focal loss for stroke prediction on fog nodes. The loss function adjusts class weights to each epoch based on inverse F1-scores, focusing more on underperforming minority cases. The DNN uses batch normalization, ReLU hidden layers, and a sigmoid output. For deployment on resource-constrained fog devices, post-training quantization compresses the float32 Keras model to a compact TFLite format.

Experiments used a stroke dataset with a 20:1 imbalance. The proposed AWFL-based DNN outperformed ANN, SVM, and random forest in AUPRC, F1, and recall. AUPRC, the key metric for imbalanced data, showed strong resilience to class imbalance. Thus, the quantized 12 KB INT8 model is ready for fog deployment.

Future work includes ensemble methods, such as stacking tree-based models with a DNN, to further boost AUPRC. The model can also be tested on other datasets, including medical images, and validated in real-time fog deployments.

DECLARATION OF COMPETING INTERESTS

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

DATA AVAILABILITY

The dataset used in this study is available at [37]. The implementation code and model training scripts are available at [38].

ACKNOWLEDGMENT

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

AI USE AND DECLARATION OF GENERATIVE AI USE

Generative AI was used to improve the tone and style of writing.

REFERENCES

[1] H.-L. Truong and S. Dustdar, "Principles for Engineering IoT Cloud Systems," *IEEE Cloud Computing*, vol. 2, no. 2, pp. 68–76, Mar. 2015, https://doi.org/10.1109/MCC.2015.23.

[2] M. Haghi Kashani, M. Madanipour, M. Nikravan, P. Asghari, and E. Mahdipour, "A systematic review of IoT in healthcare: Applications, techniques, and trends," *Journal of Network and Computer Applications*, vol. 192, Oct. 2021, Art. no. 103164, https://doi.org/10.1016/j.jnca.2021.103164.

[3] C. Li, J. Wang, S. Wang, and Y. Zhang, "A review of IoT applications in healthcare," *Neurocomputing*, vol. 565, Jan. 2024, Art. no. 127017, https://doi.org/10.1016/j.neucom.2023.127017.

[4] S. Selvaraj and S. Sundaravaradhan, "Challenges and opportunities in IoT healthcare systems: a systematic review," *SN Applied Sciences*, vol. 2, no. 1, Dec. 2019, Art. no. 139, https://doi.org/10.1007/s42452-019-1925-y.

[5] L. Hou *et al.*, "Internet of Things Cloud: Architecture and Implementation," *IEEE Communications Magazine*, vol. 54, no. 12, pp. 32–39, Dec. 2016, https://doi.org/10.1109/MCOM.2016.1600398CM.

[6] Junaid Latief Shah, Heena Farooq Bhat, and Asif Iqbal Khan, "Integration of Cloud and IoT for smart e-healthcare," in *Healthcare Paradigms in the Internet of Things Ecosystem*, Academic Press, 2021, pp. 101–136.

[7] P. Verma and S. K. Sood, "Cloud-centric IoT based disease diagnosis healthcare framework," *Journal of Parallel and Distributed Computing*, vol. 116, pp. 27–38, June 2018, https://doi.org/10.1016/j.jpdc.2017.11.018.

[8] A. V. Dastjerdi, H. Gupta, R. N. Calheiros, S. K. Ghosh, and R. Buyya, "Fog Computing: Principles, Architectures, and Applications." arXiv, Jan. 28, 2016, https://doi.org/10.48550/arXiv.1601.02752.

[9] R. K. Naha *et al.*, "Fog Computing: Survey of Trends, Architectures, Requirements, and Research Directions," *IEEE Access*, vol. 6, pp. 47980–48009, 2018, https://doi.org/10.1109/ACCESS.2018.2866491.

[10] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog Computing: A Platform for Internet of Things and Analytics," in *Big Data and Internet of Things: A Roadmap for Smart Environments*, N. Bessis and C. Dobre, Eds. Cham: Springer International Publishing, 2014, pp. 169–186.

[11] A. A. Mutlag, M. K. Abd Ghani, N. Arunkumar, M. A. Mohammed, and O. Mohd, "Enabling technologies for fog computing in healthcare IoT systems," *Future Generation Computer Systems*, vol. 90, pp. 62–78, Jan. 2019, https://doi.org/10.1016/j.future.2018.07.049.

[12] F. A. Kraemer, A. E. Braten, N. Tamkittikhun, and D. Palma, "Fog Computing in Healthcare–A Review and Discussion," *IEEE Access*, vol. 5, pp. 9206–9222, 2017, https://doi.org/10.1109/ACCESS.2017.2704100.

[13] A. Kumari, S. Tanwar, S. Tyagi, and N. Kumar, "Fog computing for Healthcare 4.0 environment: Opportunities and challenges," *Computers & Electrical Engineering*, vol. 72, pp. 1–13, Nov. 2018, https://doi.org/10.1016/j.compeleceng.2018.08.015.

[14] N. A. Mudawi, "Integration of IoT and Fog Computing in Healthcare Based the Smart Intensive Units," *IEEE Access*, vol. 10, pp. 59906–59918, 2022, https://doi.org/10.1109/ACCESS.2022.3179704.

[15] R. Priyadarshini, R. K. Barik, and H. Dubey, "DeepFog: Fog Computing-Based Deep Neural Architecture for Prediction of Stress Types, Diabetes and Hypertension Attacks," *Computation*, vol. 6, no. 4, Dec. 2018, Art. no. 62, https://doi.org/10.3390/computation6040062.

[16] Shallu, Duggal, and P. Kaur, "Assessing classification algorithms in Fog computing for diabetes diagnosis: A TOPSIS analysis of kNN, Naïve Bayes, and SVM (Cubic)," in *Smart Computing and Communication for Sustainable Convergence*, CRC Press, 2025.

[17] P. B. Corthis, G. P. Ramesh, and A. B. Jayachandra, "A Meta heuristic based deep learning classifier for effective dengue disease prediction in IoT-Fog system," *Expert Systems*, vol. 41, no. 9, 2024, Art. no. e13605, https://doi.org/10.1111/exsy.13605.

[18] Z. H. Ali, E. Hassan, S. Elgamal, and N. El-Rashidy, "Developing an explainable machine learning and fog computing-based visual rating scale for the prediction of dementia progression," *Scientific Reports*, vol. 15, no. 1, July 2025, Art. no. 25872, https://doi.org/10.1038/s41598-025-06310-4.

[19] N. A. El-Shoafy and S. I. Ghanem, "Intelligent Medical Service Monitoring Health Care System for the Elderly," in *Proceedings of the 10th International Conference on Advanced Intelligent Systems and Informatics 2024*, 2024, pp. 113–123, https://doi.org/10.1007/978-3-031-77299-3_11.

[20] Y. Gao, "Smart IoT with the hybrid evolutionary method and image processing for tumor detection," *Scientific Reports*, vol. 15, no. 1, Aug. 2025, Art. no. 31156, https://doi.org/10.1038/s41598-025-16042-0.

[21] I. Ungurean and N. C. Gaitan, "Software Architecture of a Fog Computing Node for Industrial Internet of Things," *Sensors*, vol. 21, no. 11, Jan. 2021, Art. no. 3715, https://doi.org/10.3390/s21113715.

[22] Md. S. Alom *et al.*, "Stroke Prediction Using Ensemble Machine and Deep Learning Models," *Biomedical Materials & Devices*, Oct. 2025, https://doi.org/10.1007/s44174-025-00521-z.

[23] P. Verma, R. Tiwari, W.-C. Hong, S. Upadhyay, and Y.-H. Yeh, "FETCH: A Deep Learning-Based Fog Computing and IoT Integrated Environment for Healthcare Monitoring and Diagnosis," *IEEE Access*, vol. 10, pp. 12548–12563, 2022, https://doi.org/10.1109/ACCESS.2022.3143793.

[24] S. Tuli *et al.*, "HealthFog: An ensemble deep learning based Smart Healthcare System for Automatic Diagnosis of Heart Diseases in integrated IoT and fog computing environments," *Future Generation Computer Systems*, vol. 104, pp. 187–200, Mar. 2020, https://doi.org/10.1016/j.future.2019.10.043.

[25] R. Abirami and P. E, "HAWKFOG-an enhanced deep learning framework for the Fog-IoT environment," *Frontiers in Artificial Intelligence*, vol. 7, June 2024, https://doi.org/10.3389/frai.2024.1354742.

[26] A. Pati, M. Parhi, M. Alnabhan, B. K. Pattanayak, A. K. Habboush, and M. K. Al Nawayseh, "An IoT-Fog-Cloud Integrated Framework for Real-Time Remote Cardiovascular Disease Diagnosis," *Informatics*, vol. 10, no. 1, Mar. 2023, Art. no. 21, https://doi.org/10.3390/informatics10010021.

[27] A. M. Mohamed, H. M. Amer, A. H. Rabie, A. I. Saleh, and M.-E. A. Abo-Elsoud, "Real-time monitoring system for early stroke detection based on fog computing and enhanced deep learning techniques," *Scientific Reports*, vol. 15, no. 1, Dec. 2025, Art. no. 44671, https://doi.org/10.1038/s41598-025-28513-5.

[28] S. Iftikhar, M. Golec, D. Chowdhury, S. S. Gill, and S. Uhlig, "FogDLearner: A Deep Learning-based Cardiac Health Diagnosis Framework using Fog Computing," in *Proceedings of the 2022 Australasian Computer Science Week*, Mar. 2022, pp. 136–144, https://doi.org/10.1145/3511616.3513108.

[29] M. S. Singh, K. Thongam, P. Choudhary, and P. K. Bhagat, "Stroke Risk Prediction and Prevention: Traditional versus Machine Learning Approaches," *Archives of Computational Methods in Engineering*, vol. 33, no. 3, pp. 3583–3634, Apr. 2026, https://doi.org/10.1007/s11831-025-10406-5.

[30] H. J. Suleiman, I. R. A. Hamid, and O. R. Olaniran, "Smart Health Monitoring for Predicting Heart Disease using IoT-Fog-Cloud Computing Model," *Engineering, Technology & Applied Science Research*, vol. 15, no. 3, pp. 22565–22572, June 2025, https://doi.org/10.48084/etasr.10048.

[31] S. Gupta and S. Raheja, "Stroke Prediction using Machine Learning Methods," in *2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence)*, Jan. 2022, pp. 553–558, https://doi.org/10.1109/Confluence52989.2022.9734197.

[32] N. Melnykova, Y. Patereha, S. Skopivskyi, M. Farion, S. Fedushko, and K. Drohomyretska, "Machine learning for stroke prediction using imbalanced data," *Scientific Reports*, vol. 15, no. 1, Sept. 2025, p. 33773, https://doi.org/10.1038/s41598-025-01855-w.

[33] S. Naveen and M. R. Kounte, "Optimized Convolutional Neural Network at the IoT edge for image detection using pruning and quantization," *Multimedia Tools and Applications*, vol. 84, no. 9, pp. 5435–5455, Mar. 2025, https://doi.org/10.1007/s11042-024-20523-1.

[34] K. S. Totad, A. R. Hanchinal, N. R. Shanbhog, T. V. Patgar, and P. M. Dhulavvagol, "Quantization Techniques for Optimizing MobileNetV3Large in Yoga Pose Recognition on Edge Devices," *Procedia Computer Science*, vol. 260, pp. 1000–1008, Jan. 2025, https://doi.org/10.1016/j.procs.2025.03.284.

[35] W. Samek, L. Arras, A. Osman, G. Montavon, and K.-R. Müller, "Explaining the Decisions of Convolutional and Recurrent Neural Networks," in *Mathematical Aspects of Deep Learning*, G. Kutyniok and P. Grohs, Eds. Cambridge: Cambridge University Press, 2022, pp. 229–266.

[36] N. Mahmoodi, H. Shirazi, M. Fakhredanesh, and K. DadashtabarAhmadi, "Automatically weighted focal loss for imbalance learning," *Neural Computing and Applications*, vol. 37, no. 5, pp. 4035–4052, Feb. 2025, https://doi.org/10.1007/s00521-024-10323-x.

[37] "Stroke Prediction Dataset." https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset.

[38] V. Thoday, "Vijayetha/Autho-Focal-Loss-WIth-DNN." May 13, 2026, [Online]. Available: https://github.com/Vijayetha/Autho-Focal-Loss-WIth-DNN.