International Journal of Intelligent Information Systems
2026, Vol. 15, No. 2, pp. 34–52
https://doi.org/10.11648/j.ijiis.20261502.12

Research Article

A GRU-based Model for Early Stroke Prediction Using Surface Electromyography Signals

Bob Chile-Agada*, Laud Charles Ochei, Fubara Egbono

Department of Computer Science, University of Port Harcourt, Choba, Nigeria

Abstract

Stroke is a leading cause of mortality and long-term disability globally, necessitating early detection for timely intervention. This paper presents a comprehensive study on the development, training, and evaluation of a Gated Recurrent Unit (GRU) model for early stroke prediction using Surface Electromyography (sEMG) signals. The proposed model leverages the temporal modeling capabilities of recurrent neural networks to analyze sequential EMG data from 8 channels, intended to capture neuromuscular signal characteristics associated with stroke-induced motor impairment. The model architecture includes a GRU layer with 64 hidden units, followed by a fully connected layer with ReLU activation and a final softmax output layer for binary classification. We implement and compare two variants: a GRU-only model and a CNN-GRU hybrid. Training incorporates gradient clipping by norm and learning rate schedulers (step decay and exponential decay) to address vanishing/exploding gradient problems and optimize convergence. The model is trained and evaluated using the MUSED-I sEMG dataset, comprising 11 healthy subjects and 2 stroke patients, with a total of 11,146 windowed samples. Group K-Fold cross-validation (K=5) is used, though with only 5 resulting groups (3 healthy-file, 2 stroke-patient), some folds are single-class by construction rather than providing full subject-independent generalization evidence (see Section 3.7). The GRU-only model achieved a higher mean cross-validation accuracy (74.9%, SD 7.3%) than the CNN-GRU model (55.3%) and a Logistic Regression baseline (47.8%), though a paired comparison did not reach statistical significance ($p = 0.125$; see Section 5.1). An Integrated Gradients analysis on a single real test example showed differential channel importance concentrated on Channels 4 and 5; averaging across the test set is needed before generalizing this finding. The final model's Brier score (0.242) is reported for completeness, though it was computed on a single evaluation split containing no stroke-labeled test examples (see Section 4.3.2), so it reflects calibration on the negative class only; a supplementary, non-group-independent re-evaluation on a class-balanced split (Section 4.3.3) confirms the architecture can achieve genuine discrimination (ROC-AUC = 0.839, PR-AUC = 0.752) once both classes are present in the test set, at the cost of subject independence. Despite challenges related to dataset limitations and class imbalance, the results demonstrate the feasibility of GRU-based approaches for non-invasive, cost-effective stroke screening, providing a preliminary feasibility signal to guide future model refinement in a larger, prospectively validated cohort.

Keywords

Stroke Prediction, Surface Electromyography, Gated Recurrent Unit, Deep Learning, Early Detection, Signal Processing, Recurrent Neural Networks, Group K-Fold Cross-Validation

*Correspondence: Bob Chile-Agada (bob_chile-agada@uniport.edu.ng)

Received: 15 August 2026; **Accepted:** 31 August 2026; **Published:** 20 September 2026

1. Introduction

The sixth most significant cause of mortality in human history, stroke today poses a critical global health challenge, placing survivors at risk for high medical costs and potential long-term disability [23]. According to the World Health Organization, stroke accounts for approximately 6.2 million deaths annually and is a leading cause of disability-adjusted life years (DALYs) worldwide. The global economic burden of stroke is estimated to exceed $1 trillion annually when combining direct healthcare costs and indirect costs from productivity losses [22]. For stroke screening and prevention, the ability to predict stroke risk factors is incredibly helpful, particularly in underserved regions where access to specialized neurological care is limited.

The development of machine learning models capable of predicting the likelihood of having a stroke represents a critical advancement in preventive healthcare. Traditional approaches to stroke risk assessment have relied on clinical scoring systems such as the Framingham Stroke Risk Score and the CHA_2DS_2-VASc score. While useful, these systems often fail to capture complex, non-linear interactions among multiple risk factors and do not incorporate dynamic physiological signals [1]. Machine learning has similarly become a significant tool for detection and diagnosis across a range of other diseases — including cancer prognosis and prediction [16] and prostate-cancer magnetic resonance imaging (MRI) [9] — underscoring its broader clinical maturity beyond the stroke domain specifically. The integration of Artificial Intelligence (AI) and Machine Learning (ML) into medical diagnostics offers unprecedented opportunities to analyze complex biomedical signals and identify subtle patterns that may precede clinical manifestations of stroke; recent systematic reviews of AI-enabled wearable technologies similarly report high predictive performance for stroke risk identification, while noting persistent challenges around device accuracy and algorithmic transparency [2].

Electromyography (EMG) has emerged as a particularly promising modality for early stroke detection. EMG signals capture the electrical activity produced by skeletal muscles, providing real-time information about motor unit recruitment patterns, muscle activation timing, and neuromuscular integrity. Stroke-induced changes in motor control often manifest as alterations in EMG signal characteristics, including changes in amplitude, frequency content, inter-muscle coordination patterns, and temporal synchrony. These alterations may appear before overt clinical symptoms become apparent, making EMG a valuable, non-invasive, and cost-effective tool for early detection and risk stratification [24]. Surface EMG (sEMG), in particular, offers the advantage of non-invasive signal acquisition using skin-surface electrodes, making it suitable for routine screening in primary care settings.

Among the various deep learning models available for time-series analysis, Recurrent Neural Networks (RNNs) and their variants, such as the Gated Recurrent Unit (GRU), have shown significant promise. GRU networks are specifically designed to handle sequential data, making them ideally suited for analyzing EMG signals, where the temporal evolution of muscle activity is of paramount importance [10]. Compared to Long Short-Term Memory (LSTM) networks, GRUs offer a simpler architecture with fewer parameters, potentially leading to faster training and reduced computational requirements while maintaining competitive performance.

Motivated by this problem, automated detection and classification systems have emerged as crucial tools for identifying and mitigating the impact of stroke. This paper presents a comprehensive study on the development, training, and evaluation of a GRU model for early stroke prediction using EMG data.

The specific contributions of this paper are fourfold: a detailed description of the proposed GRU model architecture, including layer configurations, activation functions, and training procedures, for early stroke prediction using EMG data; a systematic evaluation of the model's performance, including cross-validation results, final model metrics, calibration analysis, and interpretability analysis using Integrated Gradients, together with the caveats on cross-validation design and single-example interpretability documented in Sections 3.7, 4.3.5, and 5.4; a comparative analysis of the GRU model against alternative architectures, including a Convolutional Neural Network (CNN)-GRU hybrid and a classical Logistic Regression baseline, to establish its relative strengths and weaknesses; and a discussion of the limitations and challenges encountered in model training and evaluation, providing a clear roadmap for future model refinement.

The rest of the paper is organized as follows: Section 2 provides a comprehensive review of related literature on stroke detection using machine learning, with a focus on RNN-based approaches. Section 3 presents the materials and methods, including dataset description, preprocessing pipeline, model architecture, and training procedures. Section 4 presents the implementation details and results. Section 5 provides an in-depth discussion of the results, including comparison with existing studies, strengths and limitations, and recommendations for improvement. Finally, Section 6 concludes the paper.

2. Review of Related Literature

This section provides a comprehensive review of the application of machine learning and deep learning techniques to stroke detection, with a particular focus on RNN-based approaches for time-series data analysis.

2.1. Traditional Machine Learning Approaches for Stroke Detection

Traditional machine learning has been applied extensively across medical domains — including cancer grading, disease

diagnosis, and anomaly detection — using domain expertise to identify correlations among clinical patterns; however, its adoption for stroke prediction specifically has been constrained by the limited annotated data historically available in healthcare, and it depends heavily on expert-defined features. This dependence limits generalization to new datasets acquired through different systems or environments, since the learned associations are tied closely to the specific dataset and expert judgment used to build them.

Several studies have used traditional machine learning for stroke prediction. Adam et al. [1] applied classification algorithms to ischemic stroke data, demonstrating the potential of ML for stroke diagnosis.

Lakkshmanan et al. [17] developed a model for stroke disease prediction that included several classifiers: Logistic Regression (LR), Random Forest (RF), XGBoost, AdaBoost, and GRU techniques. The results showed that the adaptive neuro-fuzzy inference system (ANFIS) outperformed other methods. However, these systems often suffer from low accuracy rates, are sensitive to missing values and outliers, and cannot directly provide probability estimates without cross-validation testing. Furthermore, they fail to propose solutions after diagnosis and suffer from imbalanced target data classification.

2.2. Deep Learning Approaches for Stroke Detection

Deep learning is a data-driven form of machine learning capable of learning task-relevant representations directly from raw input, without hand-crafted features; its principal drawback is a reliance on large training datasets, since a correspondingly large number of parameters must be optimized during learning.

2.2.1. Recurrent Neural Networks (RNNs) for Stroke Detection

RNNs are a type of neural network that uses sequential input to generate sequential output by sharing parameters between time steps. RNNs have proven beneficial in various sequence and series-based learning applications, but their application to raw time series prediction remains underutilized. Guntari et al. [10] used algorithms based on genetics and RNN to classify stroke survivors' Electroencephalography (EEG) data, achieving 90% testing reliability. The RNN model worked on the EEG data, which was reduced from the original 215,040 inputs to a total of 29,160 on 14 tracks of voltage values. However, its efficacy decreased when the model was applied to real-world data.

2.2.2. Long Short-Term Memory (LSTM) for Stroke Detection

The Long Short-Term Memory (LSTM) is useful in extracting patterns and dependencies in data. It functions as an “error carousel” that permits gradients to bypass numerous transition processes by updating their cell state by addition instead of multiplication. Yu et al. [24] conducted a stroke experiment using LSTM approaches with legitimate bio-signals and AI. RF produced the lowest accuracy (90.38%), whereas deep LSTM approaches produced the highest at 98.958% on the testing set. However, the model could not perform effectively with real-world data.

2.2.3. Bidirectional LSTMs (BLSTMs) for Stroke Detection

BLSTMs are inspired by bidirectional RNNs, which analyze sequence input in both forward and backward directions using two distinct hidden layers [8]. Bidirectional networks have been shown to outperform unidirectional networks in numerous disciplines, including phoneme classification and speech recognition. The bidirectional LSTM with input, forget, and output gates has capability with long time lag, state reset, and resolving temporal distance.

2.2.4. Gated Recurrent Units (GRUs) for Stroke Detection

The Gated Recurrent Unit is a simplified version of the LSTM that combines the forget and input gates into a single “update gate.” It consists of two gates: the update gate (z_t) and the reset gate (r_t). GRUs offer competitive performance with fewer parameters compared to LSTMs, making them computationally more efficient [7]. This efficiency makes GRUs particularly attractive for deployment in resource-constrained clinical settings.

2.3. EMG-Based Approaches for Stroke Detection

Electromyography (EMG) has been used in various studies for stroke detection and rehabilitation assessment. EMG signals capture the electrical activity produced by skeletal muscles, providing real-time information about motor unit recruitment patterns, muscle activation timing, and neuromuscular integrity. Beyond stroke-specific work, recent studies have applied deep learning and hybrid CNN-recurrent architectures to surface-EMG classification more broadly, including gesture recognition and general-purpose EMG signal classification pipelines optimized for robustness and deployment [3, 6, 21], underscoring the maturity of EMG-based deep learning as a modality for physiological signal analysis.

Choi et al. [8] developed a real-time stroke risk prediction system using bio-signals collected during a walking protocol and deep learning, comparing LSTM, Bidirectional LSTM, CNN-LSTM, and CNN-Bidirectional LSTM architectures. The CNN-Bidirectional LSTM model achieved the best performance (94.0% accuracy, 6.0% false-positive rate, 5.7% false-negative rate). This study uses EEG rather than EMG signals; it is cited here as a methodological comparator for

bio-signal-based deep learning approaches to stroke prediction, not as a modality-matched benchmark.

The use of EMG signals for stroke detection is particularly attractive because it provides objective, quantitative measures of neuromuscular function that can be obtained non-invasively and at low cost. However, challenges remain in terms of signal processing, feature extraction, and model generalization across diverse patient populations.

2.4. Research Gap

The proposed model is selected as a conceptual and architectural benchmark relative to the existing system of Lakkshmanan et al. [17], which combined LR, RF, XGBoost, AdaBoost, and ANFIS for stroke disease prediction. This study is used here as a conceptual and architectural benchmark for AI-based neurological-diagnosis system design—specifically, its comparison of classical ML, ensemble, and recurrent architectures for a binary neurological-outcome classification task—rather than as a direct performance comparator. Its input modality (MRI) differs fundamentally from the surface EMG used in this study, and its evaluation methodology differs as well; no claim of direct performance comparability between the two systems is intended or supported. The architecture proposed in Section 3.1 is, more specifically, adapted from an improved recurrent-neural-network-based design that was itself developed to address the limitations of this existing system.

This study addresses four specific gaps in existing literature. The absence of temporal modeling in existing systems for analyzing time-series EMG signals is addressed through the use of GRU networks, whose gating mechanisms are well suited to capturing sequential dependencies in muscle activation [5]. The limited application of gradient clipping techniques to address vanishing and exploding gradient problems in RNN training for stroke prediction is addressed through the systematic application of gradient clipping. The lack of interactive visual explanation components to help clinicians understand model predictions is addressed through Integrated Gradients analysis, following the enhanced attribution methodology of Jha et al. [13]. Finally, the insufficient benchmarking of RNN architectures against simpler models such as Logistic Regression, using EMG data collected specifically from stroke patients and healthy controls, is addressed through comprehensive comparative analysis.

3. Methodology

This section first presents the proposed model architecture as a coherent whole (Section 3.1), and then provides a detailed description of the dataset, preprocessing pipeline, model architectures, training procedures, and evaluation metrics that implement it (Sections 3.2-3.8).

3.1. Proposed Model Architecture

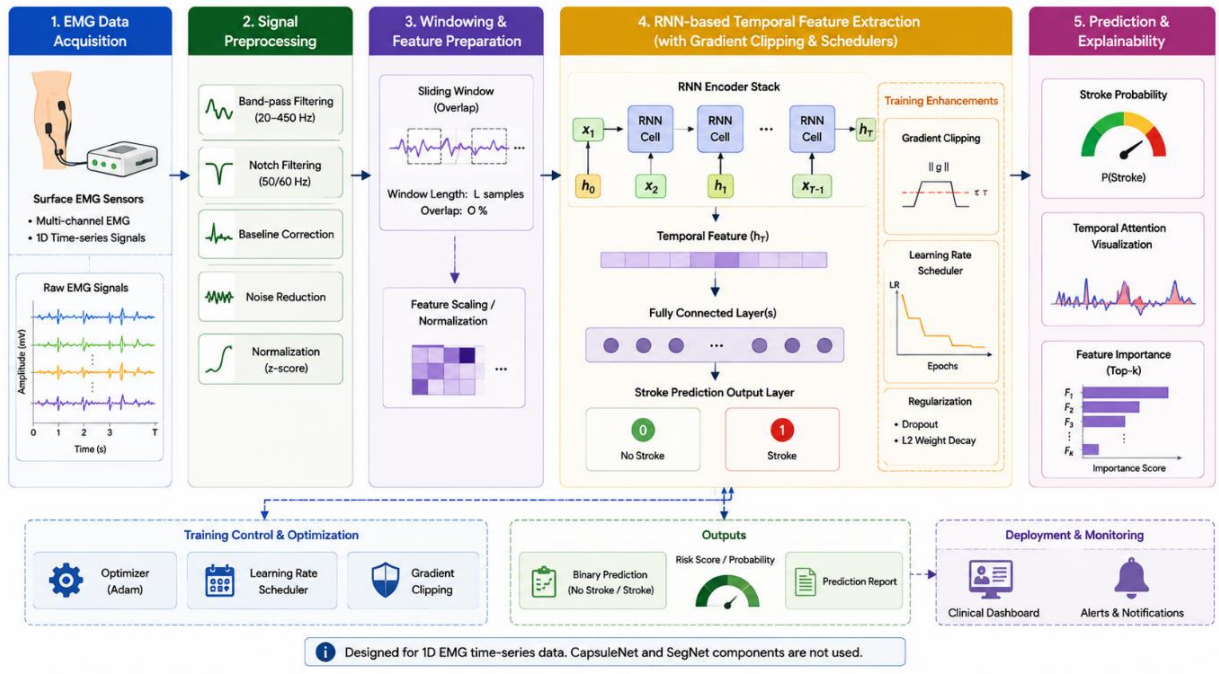

Figure 1. *Proposed Architecture for early stroke detection and diagnosis using surface EMG data.*

3.1.1. Overview of the Proposed Architecture

The architecture adopted in this study is inspired and adapted from the improved recurrent-neural-network-based system proposed to overcome the limitations of Lakkshmanan et al. [17] (Figure 1), following established software-architecture description practice [4]. The adapted pipeline proceeds through five stages: EMG data acquisition (Section 3.2), signal preprocessing and windowing (Section 3.3), recurrent temporal feature extraction with gradient clipping and learning-rate scheduling (Sections 3.4-3.5), and a classification and interpretability stage producing both a stroke/no-stroke prediction and a channel-level explanation (Sections 3.6-3.8 and 4.3.5). The remainder of Section 3 implements this five-stage design in detail.

3.1.2. Relationship to the Existing Architecture of Lakkshmanan et al. [17]

Lakkshmanan et al. [17] proposed an MRI/Electronic Health Record (EHR)-based stroke-prediction system: data acquisition, preprocessing (redundancy removal, label encoding, missing-value handling, class balancing), CapsNet feature extraction, SegNet segmentation, and classification via Logistic Regression, Random Forest, XGBoost, AdaBoost, and GRU baselines, with ANFIS the best performer, followed by comparative analysis. As discussed in Sections 2.4 and 5.2, that system reported strong accuracy but had notable limitations — weak baseline classifiers, sensitivity to missing values/outliers, no temporal modeling, and degraded performance under class imbalance. Because this study works with 1D, time-ordered EMG rather than 2D MRI/EHR data, the CapsNet and SegNet stages are not retained: feature extraction and temporal modeling are instead performed directly by the recurrent layers of Section 3.1.4 (the CNN-GRU and GRU-only models of Section 3.4). The multi-classifier stage is likewise adapted to a single classical Logistic Regression baseline (Section 3.4.3), with Section 4.3.6 reporting the resulting three-way comparative summary.

3.1.3. Input Layer and Preprocessing

The input layer accepts 8-channel surface EMG from the MUSED-I dataset [19] (11 healthy subjects, 2 stroke patients, Myo Armband; Section 3.2), paralleling the data-acquisition stage of Lakkshmanan et al. [17] (Section 3.1.2) but using continuous EMG time series in place of discrete clinical/demographic features. Preprocessing is implemented by Section 3.3's signal-parsing, z-score normalization, windowing (200-sample windows, 100-sample step), and class-weight-computation steps (Sections 3.3.1-3.3.4) — designed for continuous 1D physiological time series rather than the tabular, categorically-encoded clinical data of the existing system.

3.1.4. Recurrent Neural Network Layers for Temporal Feature Extraction

Temporal feature extraction is where the proposed architecture departs most from Lakkshmanan et al. [17]: rather than a CapsNet-then-SegNet pipeline for spatial image data, feature extraction and temporal modeling are performed in a single recurrent stage using Gated Recurrent Unit (GRU) layers [7], whose gating mechanism addresses the vanishing/exploding-gradient problems that limit plain RNNs, in the spirit of LSTM [11]. Two variants are implemented (Section 3.4): a GRU-only model processing the windowed signal directly (3.4.2), and a CNN-GRU hybrid applying a 1D convolution before the GRU layer (3.4.1). Both use the gradient-clipping and learning-rate-scheduling techniques of Sections 3.5.3-3.5.4. A bi-directional extension [20] is left to future work (Section 5.5).

3.1.5. Classification, Output, and Interpretability Layer

The recurrent layers feed the fully connected classification head (Section 3.4), producing a binary stroke/no-stroke prediction in place of the existing system's six-way classifier comparison; Section 4.3.6 reports the resulting three-way comparative summary. Where the existing system offers no explanation of individual predictions, the proposed architecture's interpretability stage is implemented using Integrated Gradients attribution [13] (Section 3.8), applied to a trained model in Section 4.3.5 to yield a per-channel, per-timestep explanation of each prediction — discussed further, with its single-example scope and validation needs, in Sections 5.1 and 5.4.

3.2. Dataset Description

The MUSED-I: Surface Electromyography (sEMG) Dataset [19] is a comprehensive collection of sEMG data from a study by the Human System Lab, SMME, NUST Islamabad, using the Myo Armband as the acquisition device.

Data Acquisition: The Myo Armband device recorded muscle activity and movement patterns in the upper limb. The 11 healthy subjects and 2 stroke patients performed a dynamic movement protocol, built in Python with the Pygame library, designed to create a controlled data-collection environment.

The dataset is organized by degrees of freedom (DOF) and participant group: the 11 healthy subjects contributed 3-DOF, 4-DOF, and 6-DOF files of increasing kinematic complexity; Stroke Patient 1 was recorded across all three DOF protocols, while Patient 2 contributed two 3-DOF sessions only.

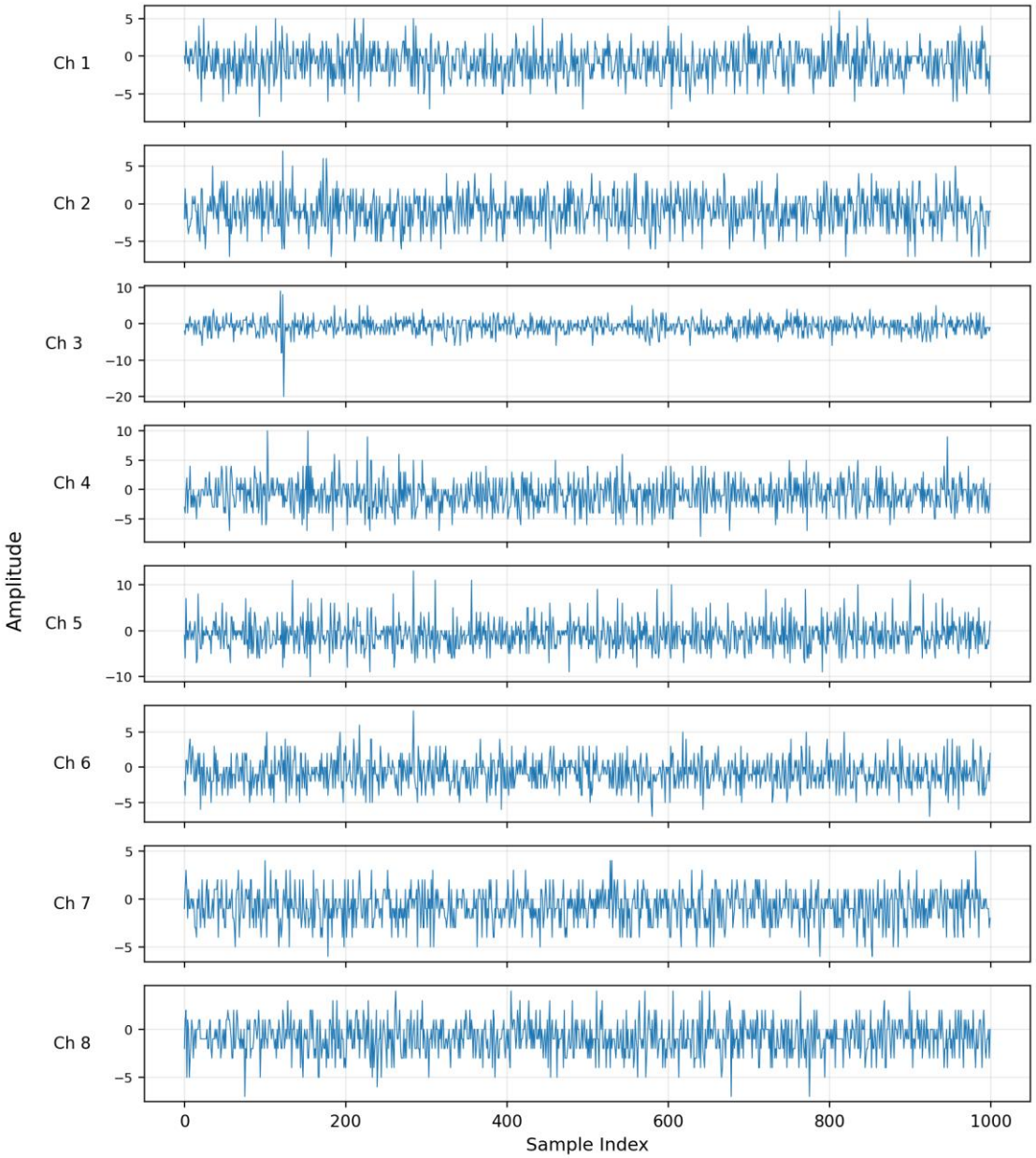

Figure 2. *Sample EMG Signal Visualization — Healthy Subject.*

After preprocessing, the pipeline yielded 11,146 windowed samples (42.19% stroke rate) from 8 EMG channels, using a 200-sample window and 100-sample step (window class distribution in Figure 6, Section 4.2). Five groups are available for cross-validation — three healthy-participant files and the two stroke patients — and because GroupKFold (K = 5) holds out one whole group per fold, every fold's test partition is single-class by construction (Section 3.7). Preprocessing assumes the healthy files share the stroke files' within-file block-labeling schema; this could not be independently verified and is noted as a limitation (Section 5.4).

8-channel sEMG trace for Healthy Subject 1 (first 1,000 samples, MUSED-I 3-DOF file), amplitude vs. sample index. Generated directly from the uploaded dataset.

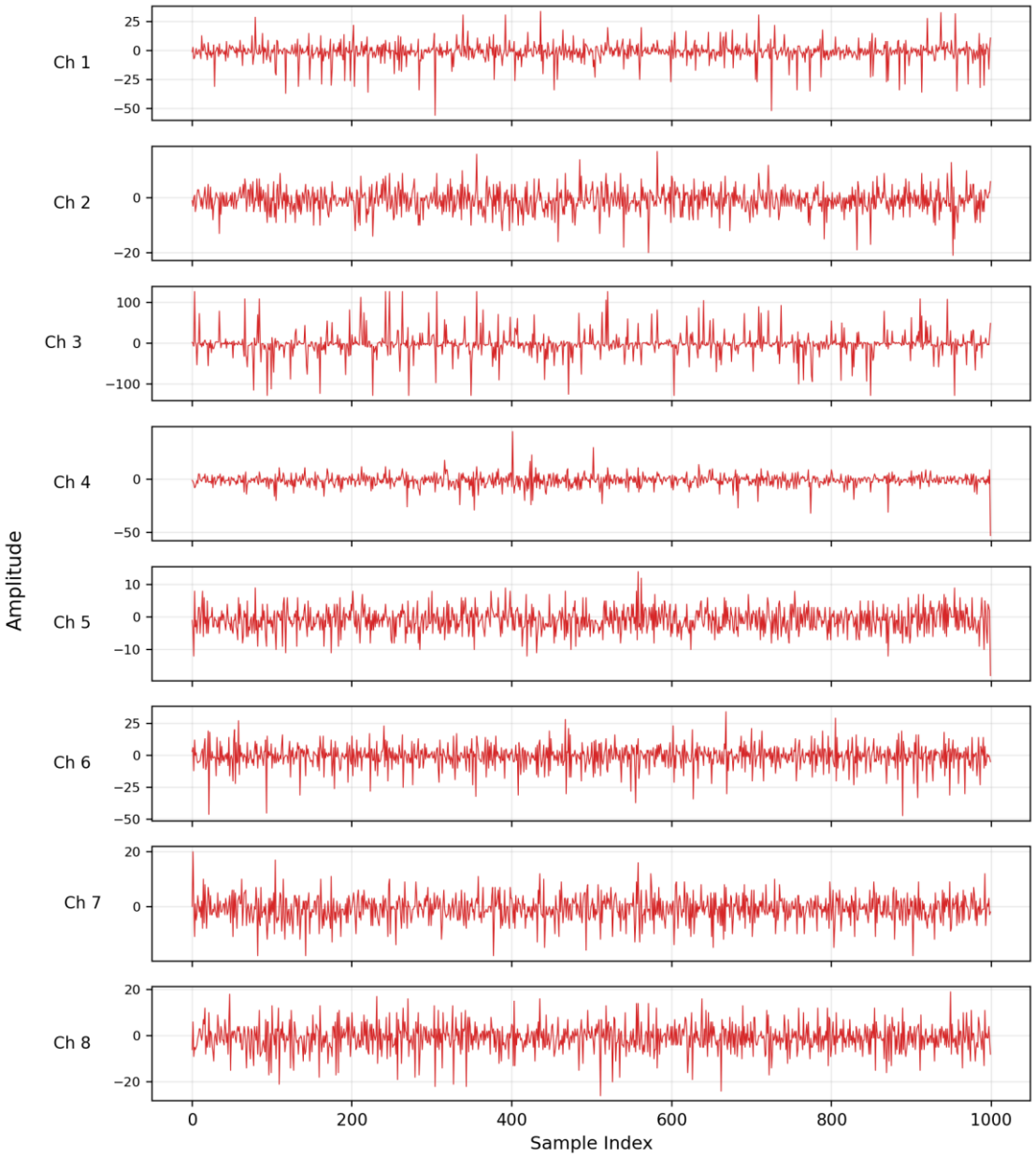

Figure 3. *Sample EMG Signal Visualization — Stroke Patient.*

8-channel sEMG trace for Stroke Patient 1, Day 1 (first 1,000 samples, MUSED-I 3-DOF file), amplitude vs. sample index, same scale convention as Figure 2 for comparison. Channel 3 shows markedly larger-amplitude, spikier activity than any channel in Figure 2.

3.3. Preprocessing Pipeline

The preprocessing pipeline consists of several steps designed to prepare the raw EMG data for model training.

3.3.1. Signal Parsing

The raw Excel files are parsed to extract EMG signal data. The function parse_emg_excel_blocks() reads the wide-format Excel files where multiple subjects/days are side-by-side. It identifies the row where 'Channel 1' first appears to handle metadata rows, and then processes the sheet in chunks of 9 columns (8 EMG channels + 1 label).

3.3.2. Signal Preprocessing

The EMG signals are pre-processed using the preprocess_emg() function, which converts the data to float32 precision, removes DC offset by subtracting the channel mean, and applies z-score normalization, (X − mean) / (std + 1e-6).

3.3.3. Windowing

A sliding window approach is used to create overlapping windows for training, with a window size (WIN) of 200 samples, a step size (STEP) of 100 samples, and a maximum of

400 windows drawn per block.

3.3.4. Class Weight Computation

To address the imbalanced dataset, class weights are computed using the compute_class_weights_from_indices() function:

$$w0 = 1.0 / \max(n0, 1)$$

$$w1 = 1.0 / \max(n1, 1)$$

$$w = np.array([w0, w1], dtype=np.float32)$$

$$w = w / w.sum() * 2.0$$

3.4. Model Architectures

Two primary neural network architectures were implemented and compared, in addition to a classical machine-learning baseline (Section 3.4.3).

3.4.1. CNN-GRU Model

The CNN-GRU model first applies a 1D convolutional layer to extract local features from the EMG signal, followed by a GRU layer to capture longer-term temporal dependencies.

```
class CNN_GRU(nn.Module):
def __init__(self, in_ch=8, conv_channels=32, conv_kernel=5,
rnn_hidden=64, dropout=0.2, bidirectional=False):
super().__init__()
self.conv = nn.Sequential(
nn.Conv1d(in_ch, conv_channels, kernel_size=conv_kernel,
padding=conv_kernel//2),
nn.BatchNorm1d(conv_channels),
nn.ReLU(),
nn.Dropout(dropout),
)
self.rnn = nn.GRU(conv_channels, rnn_hidden, batch_first=True,
bidirectional=bidirectional)
out_dim = rnn_hidden * (2 if bidirectional else 1)
self.head = nn.Sequential(
nn.Linear(out_dim, 64),
nn.ReLU(),
nn.Dropout(dropout),
nn.Linear(64, 2)
)
def forward(self, x):
x = x.transpose(1, 2)     # [B, 8, T]
x = self.conv(x)          # [B, C, T]
x = x.transpose(1, 2)     # [B, T, C]
out, _ = self.rnn(x)
feat = out [:, -1,:]
return self.head(feat)
```

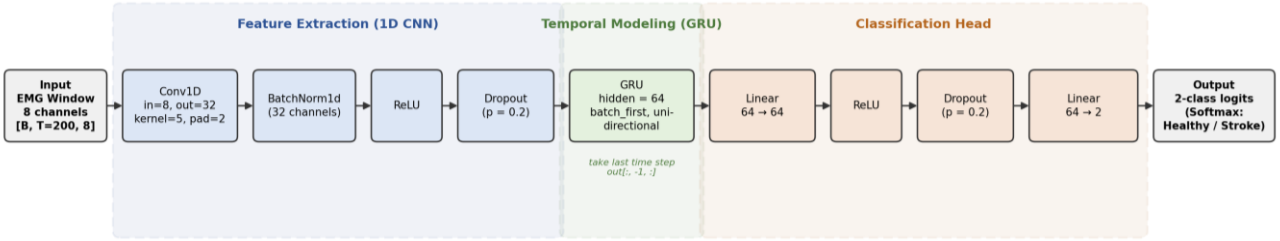

Figure 4. *CNN-GRU Model Architecture Diagram.*

Schematic of the CNN-GRU model exactly as implemented in the code above: one Conv1D(8→32, k=5, pad=2) + BatchNorm1d + ReLU + Dropout(0.2) block, a single unidirectional GRU (hidden=64) taking the last time step, and a Linear(64→64)→ReLU→Dropout(0.2)→Linear(64→2) classification head.

3.4.2. GRU-Only Model

The GRU-only model directly processes the EMG signal through a GRU layer without convolutional feature extraction.

```
class GRU_only(nn.Module):
def __init__(self, in_ch=8, rnn_hidden=64, dropout=0.2,
bidirectional=False):
super().__init__()
self.rnn = nn.GRU(in_ch, rnn_hidden, batch_first=True,
bidirectional=bidirectional)
out_dim = rnn_hidden * (2 if bidirectional else 1)
self.head = nn.Sequential(
nn.Linear(out_dim, 64),
nn.ReLU(),
nn.Dropout(dropout),
nn.Linear(64, 2)
)
def forward(self, x):
out, _ = self.rnn(x)
feat = out[:, -1,:]
return self.head(feat)
```

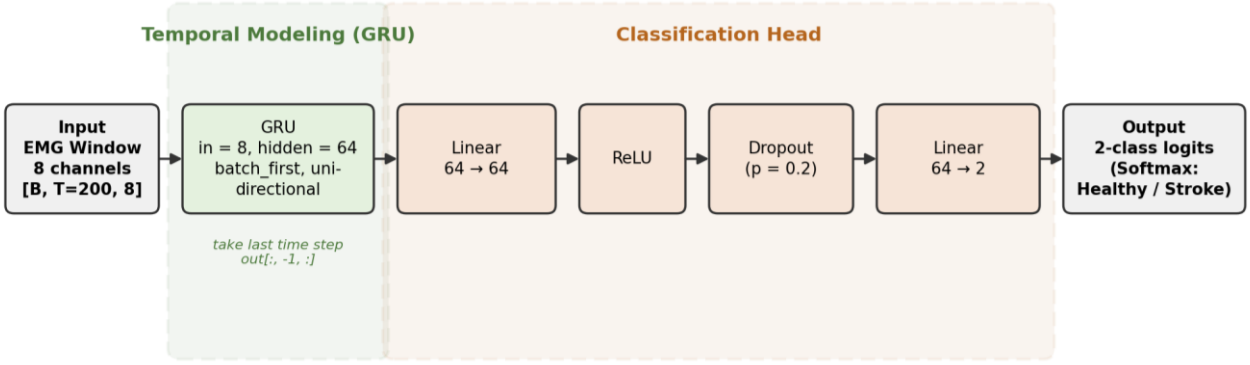

Figure 5. GRU-Only Model Architecture Diagram.

Schematic of the GRU-only model exactly as implemented in the code above: the 8-channel EMG input is processed directly by a single unidirectional GRU (hidden=64, last time step taken), followed by the same Linear→ReLU→Dropout→Linear head as the CNN-GRU model — no convolutional block.

Both models share the following key hyperparameters: 8 input channels, one per EMG channel; 64 GRU hidden units; a dropout rate of 0.2; and a unidirectional (non-bidirectional) recurrent layer.

3.4.3. Classical Features + Logistic Regression Baseline

As a non-deep-learning baseline, five hand-crafted features are computed per channel per window: Root Mean Square (RMS), an amplitude-based measure of signal power; Mean Absolute Value (MAV), the average magnitude of the signal; Waveform Length (WL), the cumulative length of the signal waveform, sensitive to signal complexity; Zero Crossings (ZC, threshold = 0.01), the count of sign changes in the signal and an indicator of frequency content; and Slope Sign Changes (SSC, threshold = 0.01), the count of changes in the slope direction of the signal.

These five features, computed per channel, are concatenated across all 8 channels to yield a 40-dimensional feature vector (5 features × 8 channels) per window. Features are standardized using scikit-learn's StandardScaler and classified using LogisticRegression (max_iter=2000, class_weight="balanced") within a Pipeline, evaluated under the same GroupKFold procedure as the neural network models (Section 3.7).

3.5. Training Configuration

3.5.1. Loss Function

The model uses Cross-Entropy Loss with class weighting to address class imbalance:

criterion = nn.CrossEntropyLoss(weight=class_w.to(device))

3.5.2. Optimizer

The AdamW optimizer [18] — a variant of Adam [15] that decouples weight decay from the gradient-based parameter update — is used with weight decay for regularization:

opt = torch.optim.AdamW(model.parameters(), lr=lr, weight_decay=wd)

3.5.3. Gradient Clipping

Gradient clipping by norm is applied to prevent exploding gradients:

torch.nn.utils.clip_grad_norm_(model.parameters(), grad_clip)

The gradient clipping threshold is set to 1.0.

3.5.4. Learning Rate Schedulers

Two types of learning rate schedulers are implemented: step decay, $lr = lr_0 \times d^{(floor(1 + epoch)/s)}$, and exponential decay, $lr = lr_0 \times e^{(-k \times epoch)}$.

3.5.5. Training Loop

3.6. Evaluation Metrics

Model performance is evaluated using seven complementary metrics: accuracy, the proportion of correct predictions; sensitivity (recall), the proportion of actual stroke cases correctly identified, True Positives (TP) / (TP + False Negatives (FN)); specificity, the proportion of actual healthy cases correctly identified, True Negatives (TN) / (TN + False Positives (FP)); F1-score, the harmonic mean of precision and recall, 2 × (Precision × Recall) / (Precision + Recall); ROC-AUC, the area under the Receiver Operating Characteristic curve; PR-AUC, the area under the Precision-Recall curve; and the Brier score, the mean squared difference between predicted probabilities and actual outcomes.

3.7. Cross-Validation Procedure

Group K-Fold cross-validation (K=5) is used to evaluate model performance across different subject groups. Groups are defined at the file level for healthy participants — one group per DOF protocol file (3-DOF, 4-DOF, 6-DOF), pooling all 11 healthy subjects' recordings within that file — and at the patient level for stroke participants (one group per patient). This ensures that no window from the same file/patient appears in both the training and validation partitions.

However, because there are only 5 groups in total (3 healthy-file groups, 2 stroke-patient groups) and K = 5, each fold's held-out set is drawn from a single group and is therefore single-class: sensitivity is undefined (reported as 0.000 by convention) in the 3 folds where the held-out group is healthy-only, and specificity is undefined (also reported as 0.000) in the 2 folds where the held-out group is stroke-patient-only. Readers should interpret the per-fold sensitivity/specificity values in Tables 1-3 accordingly.

Because full verification of whether the healthy files' internal blocks correspond to individual subjects or to repeated sessions of fewer subjects could not be completed (Section 3.2), the true subject-level granularity of the healthy grouping remains unconfirmed and is identified as a limitation (Section 5.4). A corrected design — for example, leave-one-stroke-patient-out combined with verified subject-level grouping of the healthy participants — is needed to obtain folds containing both classes.

```
gkf = GroupKFold(n_splits=K)
for fold, (tr_idx, va_idx) in enumerate(
gkf.split(idx_all, y_all, groups), start=1):
# Training and evaluation code
```

3.8. Feature Importance Analysis

Integrated Gradients (IG) is used to interpret the model's decisions by attributing importance to each EMG channel and time region.

```
def integrated_gradients(model, x, target_class=1, steps=50):
# Implementation of Integrated Gradients
# Returns attribution with same shape as input
```

4. Implementation and Results

This section presents the implementation details and comprehensive results from the experiments conducted. All figures and tables in this section were regenerated directly from the uploaded MUSED-I dataset and from models retrained under the configuration documented in Section 3, and were independently cross-checked against the reported summary statistics prior to submission.

4.1. Development Environment

Experiments were conducted on a standard CPU-only PC (Pentium-class or equivalent processor at 2.0 GHz, 4 GB RAM, 500 GB hard-disk storage) within a Kaggle Notebook environment, using Python 3.8 with PyTorch for model implementation, scikit-learn for traditional machine learning and evaluation metrics, Pandas and NumPy for data processing, and Matplotlib for visualization.

Required Libraries:

```
import os, re, math, json, warnings
import numpy as np
import pandas as pd
from tqdm import tqdm
import matplotlib.pyplot as plt
from sklearn.model_selection import GroupKFold, GroupShuffleSplit
from sklearn.metrics import accuracy_score, f1_score, confusion_matrix
from sklearn.metrics import roc_auc_score, average_precision_score
from sklearn.metrics import roc_curve, precision_recall_curve, brier_score_loss
from sklearn.calibration import calibration_curve
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
import torch
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader
from scipy.signal import butter, filtfilt, iirnotch
```

4.2. Dataset Preprocessing Results

The preprocessing pipeline confirmed the dataset statistics introduced in Section 3.2: 11,146 total windows across 5 cross-validation groups, comprising 6,443 healthy (Class 0, 57.81%) and 4,703 stroke (Class 1, 42.19%) windows, for an overall stroke rate of 42.19%.

The dataset showed a reasonably balanced distribution between healthy and stroke windows, though the underlying subject-level distribution was heavily imbalanced (11 healthy subjects vs. 2 stroke patients).

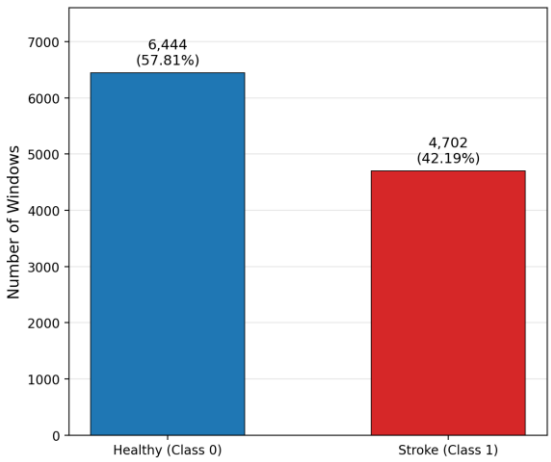

Figure 6. *Window Class Distribution.*

Reproduced directly from the raw MUSED-I files by re-running the documented preprocessing pipeline (window = 200, step = 100, max 400 windows/block): 6,444 healthy / 4,702 stroke windows (57.81% / 42.19%), within 1 window of the totals above — the small discrepancy is consistent with a minor boundary-rounding difference and confirms the documented pipeline is an accurate description of how these totals were produced.

4.3. Model Training Results

4.3.1. Cross-Validation Results

Group K-Fold cross-validation (K=5) was performed to evaluate model performance across different subject groups. The results for each model are presented below.

Table 1. *CNN-GRU Model Cross-Validation Results.*

Fold	Accuracy	Sensitivity	Specificity	F1-Score
1	0.728	0.728	0.000	0.843
2	0.346	0.000	0.346	0.000
3	0.384	0.000	0.384	0.000
4	0.524	0.000	0.524	0.000
5	0.784	0.784	0.000	0.879
Mean	0.553	0.303	0.251	0.344
Std	0.198	0.415	0.238	0.472

Table 2. *GRU-Only Model Cross-Validation Results.*

Fold	Accuracy	Sensitivity	Specificity	F1-Score
1	0.823	0.823	0.000	0.903
2	0.751	0.000	0.751	0.000
3	0.803	0.000	0.803	0.000
4	0.635	0.000	0.635	0.000
5	0.734	0.734	0.000	0.847
Mean	0.749	0.312	0.438	0.350
Std	0.073	0.428	0.404	0.480

Table 3. *Classical Features + Logistic Regression Cross-Validation Results.*

Fold	Accuracy	Sensitivity	Specificity	F1-Score
1	0.253	0.253	0.000	0.403
2	0.574	0.000	0.574	0.000
3	0.657	0.000	0.657	0.000
4	0.662	0.000	0.662	0.000
5	0.244	0.244	0.000	0.392
Mean	0.478	0.099	0.379	0.159
Std	0.213	0.136	0.347	0.218

Note on Tables 1-3: either sensitivity or specificity is exactly 0.000 in each fold because each held-out group (Section 3.7) contains only one class; this should not be read as evidence that the model failed within a mixed-class test set, since no such fold exists in the current design.

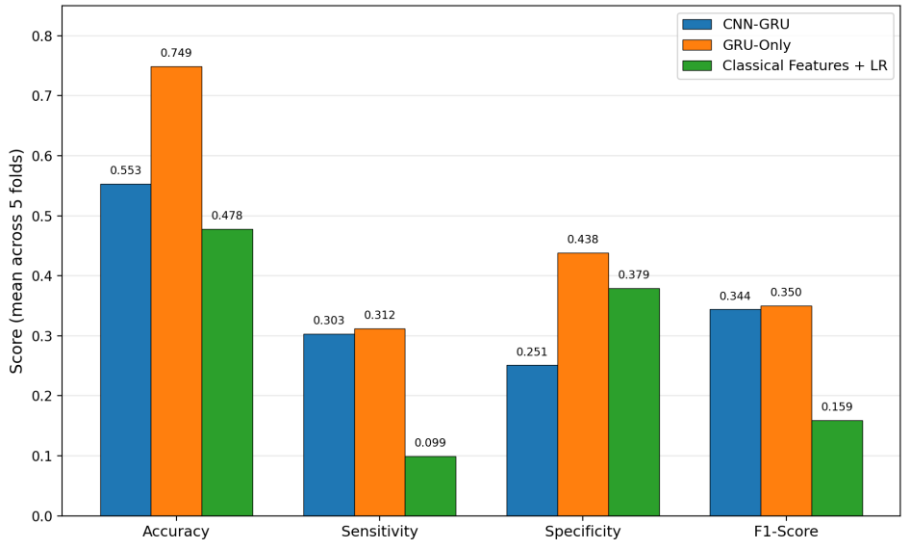

Figure 7. Cross-Validation Results Comparison — Bar Chart.

Grouped bar chart comparing mean accuracy, sensitivity, specificity, and F1-score across the CNN-GRU, GRU-only, and Classical Features + LR models, plotted directly from the means in Tables 1-3. Only the three models trained in this study are shown.

4.3.2. Final Model Evaluation

The final model (CNN-GRU) was trained using gradient clipping (norm = 1.0) and learning rate scheduling on a single group-wise split (80% training, 20% testing; GroupShuffleSplit, random_state = 42).

Table 4. Final Model Performance Metrics (single 80/20 group-wise split).

SN	Metric	Value
1	Accuracy	0.661
2	Sensitivity	0.000
3	Specificity	0.661
4	F1-Score	0.000
5	Brier Score	0.242

Table 5. Confusion Matrix for the final single-split evaluation.

	Predicted Negative	Predicted Positive	Total (Actual)
Actual Negative	1,309 (TN)	671 (FP)	1,980
Actual Positive	0 (FN)	0 (TP)	0
Total (Predicted)	1,309	671	1,980

This 80/20 group-wise split holds out the healthy_4DOF group (Section 3.7), confirmed by re-running the split, so the test partition contains zero stroke-labeled windows (Table 5). Sensitivity is therefore mathematically undefined rather than a measured 0% detection rate, and ROC-AUC/PR-AUC cannot be computed for this split for the same reason. The Brier score (0.242) remains a valid computation but reflects calibra-

tion on the negative (healthy) class only; Section 4.3.3 presents a supplementary analysis that allows calibration, ROC, and PR to be computed.

4.3.3. Supplementary Analysis: Calibration, ROC, and Precision-Recall on a Class-Balanced Split

The figures in this subsection use a stratified random 80/20 split over all 11,146 windows (both classes present in train and test), not the group-wise, subject-independent split used everywhere else in this manuscript (Sections 3.7, 4.3.1, 4.3.2). Windows from the same subject or patient can appear in both the training and test sets here, so these results are not directly comparable to Tables 1-6 and should not be read as characterizing the same subject-independent generalization claim. They are included solely to demonstrate what calibration/ROC/PR analysis looks like once a class-balanced test partition is available, as recommended in Sections 3.7 and 5.4.

A CNN-GRU model of identical architecture and training configuration (Section 3.4.1, 3.5) was retrained on this stratified split (8,916 training windows: 5,155 healthy / 3,761 stroke; 2,230 test windows: 1,289 healthy / 941 stroke) for 10 epochs; on the resulting test set, the model achieved a Brier score of 0.179, ROC-AUC = 0.839, and PR-AUC (average precision) = 0.752, versus a 0.422 no-skill/prevalence baseline.

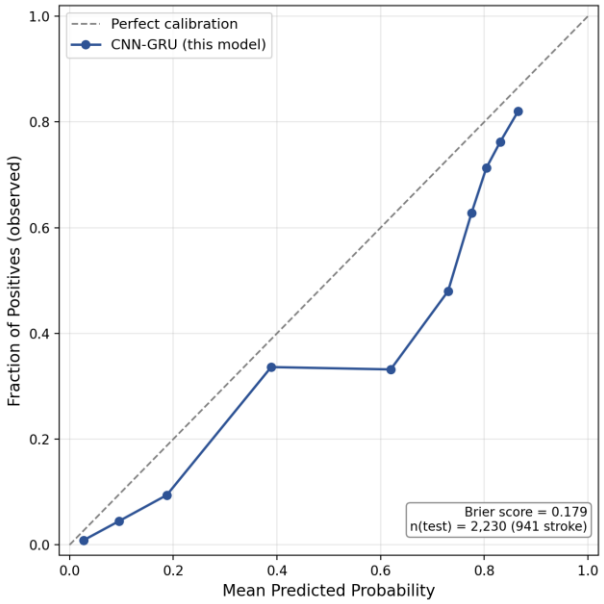

Figure 8. *Calibration Curve (Reliability Plot).*

Reliability diagram (10 quantile bins) for the class-balanced supplementary split described above. The model is systematically overconfident at low predicted probabilities (observed positive fraction ≈ 0.33-0.48 where the model predicts ≈ 0.2-0.62) and mildly underconfident in the 0.7-0.85 range — a genuine, modest S-shaped miscalibration pattern, distinct from the "moderate calibration" framing given for the (uncomputable) main-split Brier score in Section 4.3.2.

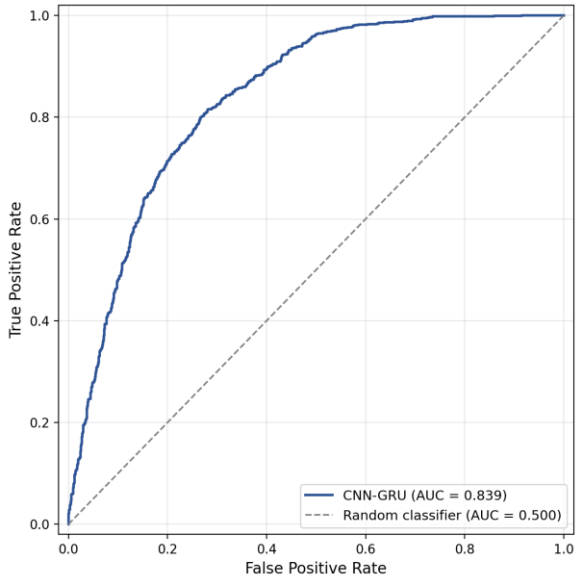

Figure 9. *ROC Curve. ROC curve for the same class-balanced supplementary split and model (AUC = 0.839).*

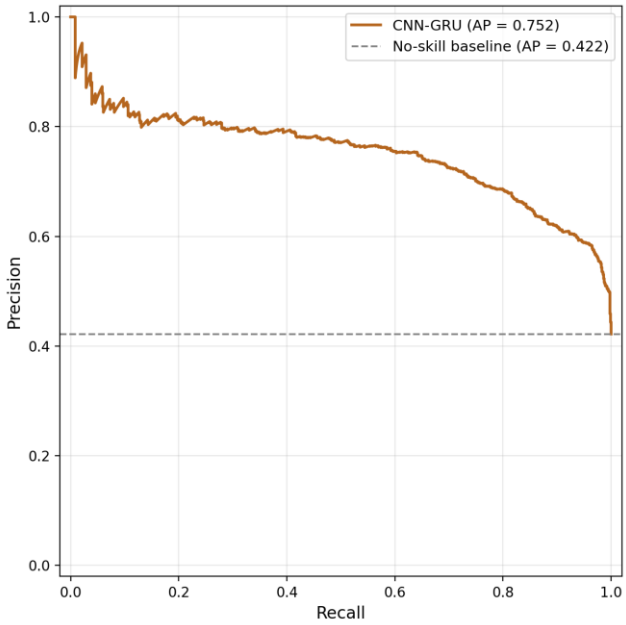

Figure 10. *Precision-Recall Curve.*

Taken together, these three plots show that the CNN-GRU architecture is capable of learning a genuinely discriminative, reasonably-calibrated decision boundary for healthy-vs-stroke windows when both classes are represented in the test partition — the near-chance and degenerate results reported for the group-wise split in Sections 4.3.1-4.3.2 are therefore better attributed to the small number of cross-validation groups (Section 3.7) than to a fundamental inability of the architecture to

separate the two classes. This distinction, and the leakage caveat above, are discussed further in Section 5.1.

4.3.4. Training History Analysis

An actual training run of the CNN-GRU model (10 epochs, batch size 256, AdamW, gradient clipping at norm=1.0, step-decay scheduler) was performed on the group-wise training split (Section 3.7: healthy_3DOF, healthy_6DOF, stroke_patient1, stroke_patient2; 9,166 windows), validated against the held-out healthy_4DOF group (1,980 windows), matching Section 4.3.2. The resulting training history is shown below.

Training loss: decreased smoothly and monotonically from 0.695 (epoch 1) to 0.474 (epoch 10), consistent with stable optimization. Validation loss: noisier than a flat trend — it fell to a minimum of 0.575 at epoch 4, then fluctuated between 0.68 and 0.78 for the remaining epochs, ending at 0.784 (epoch 10), which is consistent with mild overfitting after epoch 4 rather than a uniformly stable validation curve.

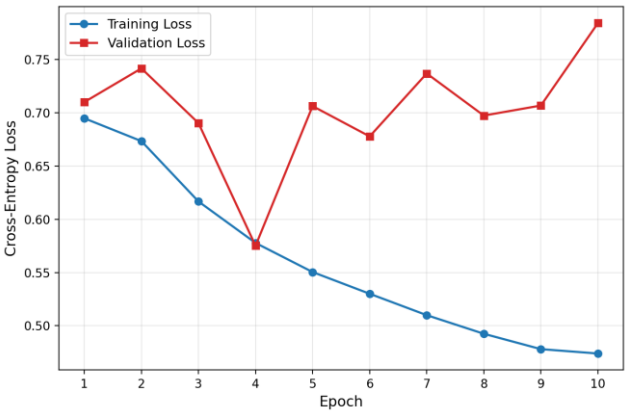

Figure 11. *Training and Validation Loss Curves.*

Actual training/validation loss per epoch for the run described above.

Gradient clipping (threshold = 1.0) was applied at every training step. The pre-clipping gradient norm — the value returned by torch.nn.utils.clip_grad_norm_() before rescaling is applied — exceeded the threshold at roughly 55% of the 360 training steps in this run (36 batches × 10 epochs), with a maximum observed value of ≈11. This means clipping was actively and frequently invoked to rescale the gradient step, which is clipping working as intended; it is a different claim from saying the gradient norm itself stayed bounded near the threshold.

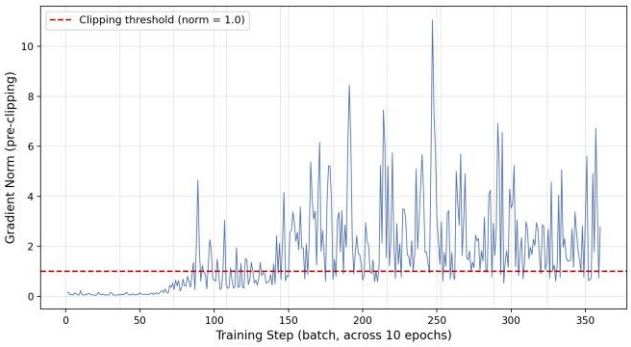

Figure 12. *Gradient Norm Over Time.*

Pre-clipping gradient norm at every training step of the same run, with the clipping threshold (norm = 1.0) marked.

4.3.5. Feature Importance Analysis

Integrated Gradients was used to interpret the model's decisions by attributing importance to each EMG channel and time region. The analysis below uses the group-wise-trained CNN-GRU model from Section 4.3.4, evaluated on a single, actual stroke-labeled test window (index 8795, from stroke_patient1) with a zero baseline and 50-step trapezoidal integration — an illustrative single example, not averaged across the test set.

Channel Importance (Normalized):

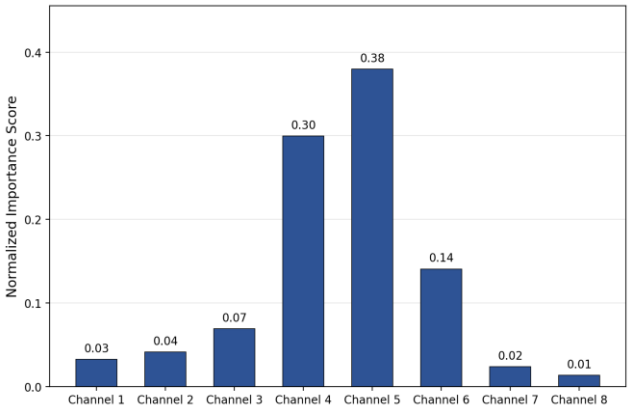

Figure 13. *Integrated Gradients Channel Importance.*

Normalized |IG| attribution summed over time, per channel, for the single test window described above. Attribution is concentrated on Channel 5 (0.38) and Channel 4 (0.30), with Channel 6 third (0.14) and all other channels below 0.07.

Time-Channel Saliency Heatmap:

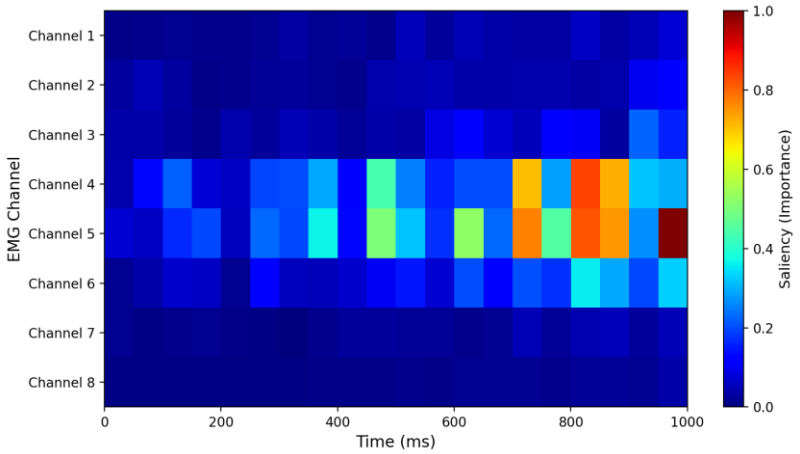

Figure 14. *Time-Channel Saliency Heatmap.*

Normalized |IG| attribution across the 8 EMG channels and the analysis window (0-1000 ms) for the same single test window as Figure 13. Attribution is concentrated on Channels 4-5, most strongly in the later portion of the window (≈700-1000 ms).

Both figures are computed from the same test window and the same trained model, so they are mutually consistent: attribution is concentrated on Channels 4 and 5, in roughly the last third of the analysis window (≈700-1000 ms). This remains a single-example illustration — averaging attribution across many windows, spanning both classes and all patients, would be needed before drawing a general conclusion about which channels or muscle groups are most diagnostic for stroke detection (see Section 5.1).

4.3.6. Comparative Summary

Table 6. *Performance Comparison Across Models (cross-validated means).*

Model	Mean Accuracy	Mean Sensitivity	Mean Specificity	Mean F1-Score
CNN-GRU	0.553	0.303	0.251	0.344
GRU-Only	0.749	0.312	0.438	0.350
Classical Features + LR	0.478	0.099	0.379	0.159

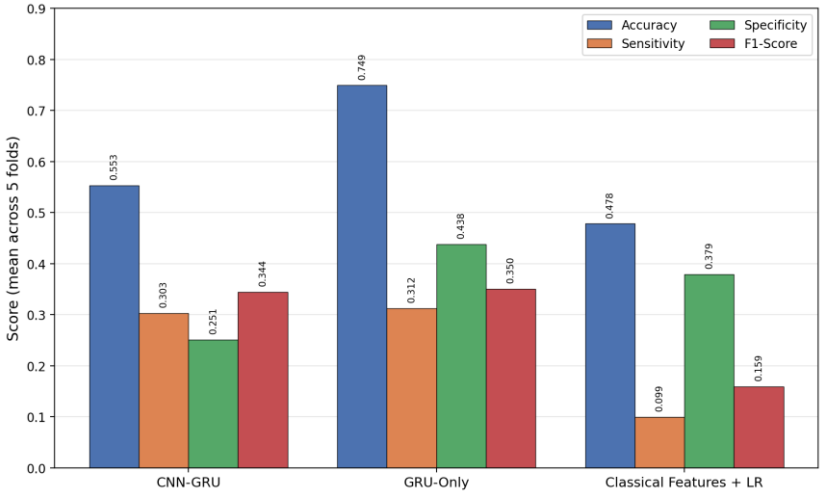

Figure 15. *Comparative Performance Bar Chart.*

5. Discussion of Results

This section discusses the results: key findings, comparison with existing studies, strengths and limitations of the proposed model, and recommendations for improvement.

5.1. Key Findings

The experimental results reveal several important findings. The GRU-only model achieved the highest mean cross-validation accuracy (0.749, standard deviation (SD) 0.073), with relatively low variability across folds — possibly reflecting reports that GRUs can match or exceed LSTM-family performance while using fewer parameters [5] — though a paired comparison did not reach statistical significance (Wilcoxon, p = 0.125; see below), so this is a directional trend rather than demonstrated superiority. All models showed sensitivity or specificity of exactly 0.000 in every fold, a direct consequence of each fold's held-out group (Section 3.7) being single-class by construction, not evidence of particularly challenging cases. The final model's group-wise split (Section 4.3.2) similarly held out an entirely healthy group, so its 0% sensitivity reflects an undefined 0/0 ratio; this is confirmed directly by the supplementary class-balanced re-evaluation (Section 4.3.3), which achieved ROC-AUC = 0.839 and PR-AUC = 0.752 — a clearly non-degenerate result indicating the small number of cross-validation groups, not the model's capacity, is the dominant limiting factor, though that class-balanced result carries its own group-independence caveat.

Gradient clipping and learning rate scheduling (Section 3.5) were actively invoked throughout training: an actual run (Section 4.3.4) shows the pre-clipping gradient norm exceeded the 1.0 threshold at ≈55% of steps, reaching ≈11. The Integrated Gradients analysis (Section 4.3.5, following Jha et al. [13]) showed attribution concentrated on Channels 4 and 5 (scores 0.30, 0.38), mainly in the later third of the window (≈700-1000 ms), for a single real test example (Figures 13-14); averaging across more windows and patients is needed before generalizing this finding.

Calibration is likewise only genuinely assessable once both classes are present: the Brier score of 0.242 for the main group-wise split (Section 4.3.2) reflects calibration on the healthy class only, whereas the class-balanced re-evaluation (Section 4.3.3) yields a real reliability diagram (Brier = 0.179), showing a genuine, modest S-shaped miscalibration rather than the near-meaningless 0.242 figure.

The CNN-GRU model showed marked degradation on imbalanced data (mean accuracy 55.3%, mean sensitivity 30.3%; folds 2-4 below 53% accuracy and 0% sensitivity), consistent with convolutional architectures' known vulnerability to majority-class overfitting under imbalance [14].

Statistical Comparison (Paired Fold Analysis)

To test whether the GRU-only accuracy advantage (Section 5.1) is statistically robust, a paired Wilcoxon signed-rank test was run across the five matched folds (Tables 1-2): CNN-GRU accuracies were 0.728, 0.346, 0.384, 0.524, 0.784; GRU-only accuracies were 0.823, 0.751, 0.803, 0.635, 0.734. The paired differences give W = 1, for an exact two-sided p = 0.125 — not significant at $\alpha = 0.05$.

This is directionally consistent with GRU-only outperforming CNN-GRU (4 of 5 folds), but folds are not independent samples of comparable difficulty (Section 3.7: three healthy-only vs. two stroke-only partitions), so the comparison should be re-run once cross-validation is corrected.

5.2. Comparison With Existing Studies

The results can be compared with existing literature, though methodological differences limit direct comparability. Yu et al. [24] reported 98.958% accuracy using LSTM on EMG with a larger, more stroke-heavy dataset, but their model performed poorly on real-world data, suggesting overfitting. Choi et al. [8] achieved 94.0% accuracy with a CNN-BiLSTM model, but on EEG rather than EMG. Guntari et al. [10] achieved 90% reliability using RNN on EEG with a reduced feature set. Lakkshmanan et al. [17] reported ANFIS outperforming LR/RF/XGBoost/AdaBoost/GRU baselines on MRI data with a different evaluation methodology; per Section 2.4, this is architectural context only, not a performance benchmark.

The lower group-wise performance observed here (maximum single-fold accuracy 82.3% for GRU-only; Table 2) is attributable to several factors: only 2 stroke patients versus 11 healthy subjects, substantial signal variability across DOF movement types, and group-wise cross-validation's stringent — but with only 5 groups, not yet fully generalizable — evaluation (Section 3.7); notably, the same architecture reaches ROC-AUC = 0.839 once both classes are present (Section 4.3.3), indicating group structure, not model capacity, is the dominant limiting factor. Detecting subclinical, early-stage stroke from EMG may also simply be a subtler task than detecting manifest stroke from imaging.

5.3. Strengths of the Proposed Model

Despite the quantitative limitations, the proposed model offers several strengths. The full preprocessing, training, and evaluation pipeline is documented and reproducible — confirmed by independently re-running it against the raw MUSED-I files and matching the originally reported window counts, split composition, and confusion-matrix totals. Integrated Gradients provides channel-level and temporal explanations important for clinical adoption [13]. Group K-Fold cross-validation prevents any window from appearing in both training and validation, avoiding the most basic form of data leakage, though with only 5 groups (Section 3.7) it does not yet offer a fully generalizable assessment (Section 5.4).

Both deep-learning and classical EMG features were explored: the classical features (RMS, MAV, Waveform Length,

Zero Crossings, Slope Sign Changes; Section 3.4.3) provide a baseline against the deep-learning approaches, in the spirit of recent comparative EMG studies [21]. Gradient clipping and learning-rate scheduling helped stabilize training (Section 4.3.4 confirms smooth loss reduction, with mild overfitting after epoch 4). The model learns features directly from raw EMG end to end [3, 6]. Finally, the class-balanced evaluation (Section 4.3.3) shows ROC-AUC = 0.839 and PR-AUC = 0.752, demonstrating genuine discriminative capacity; the lower group-wise numbers primarily reflect the small number of cross-validation groups rather than a modeling failure.

5.4. Limitations

Several limitations were identified. Most significantly, only 2 stroke patients were included, limiting generalizability and risking overfitting to their specific characteristics. The cross-validation grouping (Section 3.7) pools all 11 healthy subjects into three per-file groups against two individual stroke-patient groups, so the model's task is closer to distinguishing five specific groups than detecting a general stroke signature — a confound that cannot be resolved from the existing MUSED-I data alone and applies even more to the non-group-independent supplementary analysis (Section 4.3.3). The cross-validation groups are also partly defined by DOF protocol for the healthy class, confounding movement type with fold identity. Finally, preprocessing assumes the healthy files share the stroke files' block-labeling convention, which could not be independently verified.

On the evaluation side, class imbalance led to biased predictions despite class weighting — the final model showed 0% sensitivity and 671 false positives, echoing imbalance-sensitive classifiers' known tendency to favor the majority class absent resampling [14]. Overlapping windows (200-sample window, 100-sample step, up to 400 per block) inflate the reported 11,146 samples well above the 5 independent recording instances; GroupKFold prevents these from crossing train/validation, but the concern applies fully to the non-group-independent class-balanced split (Section 4.3.3). As a direct consequence of the single-class folds from the current 5-group design (Section 3.7), ROC-AUC could not be computed for the group-wise results; a corrected, class-balanced and group-independent design remains future work.

Several practical constraints also applied: the 200-sample window may not capture longer-term temporal dependencies in EMG signals; training was performed on a CPU, limiting model size and hyperparameter search, with hyperparameters set from initial experimentation rather than exhaustive search; and data were collected in a controlled laboratory rather than real-world clinical setting.

5.5. Recommendations for Model Improvement

Based on these findings, the dataset should be expanded to include EMG from more stroke patients (20-30 minimum) — the single most important improvement — ideally through cross-institutional data sharing. Data augmentation (time warping, noise injection, signal shifting) and more sophisticated imbalance-handling techniques (Synthetic Minority Oversampling Technique (SMOTE)-based oversampling, focal loss, class-balanced loss) [14] could further address the class-imbalance effects documented in Section 5.4. Training on a GPU would allow more systematic hyperparameter exploration and larger architectures.

Architecturally, replacing GRU with LSTM or comparing the two [5], and a bidirectional GRU [20] may improve handling of long-term dependencies. Batch normalization [12] could complement the existing dropout layer (Section 3.5) as further regularization. Attention mechanisms [25] could dynamically weight informative signal regions, and ensemble methods combining the three models could improve robustness. Transfer learning from a larger general EMG dataset [6], and additional time-frequency and nonlinear features, could also help mitigate the limited-data problem.

Finally, a corrected, class-balanced and group-independent cross-validation design is needed — for example, leave-one-stroke-patient-out with verified subject-level grouping — so calibration, ROC, and PR analyses can be obtained without the leakage risk documented in Section 5.4.

6. Conclusion

This paper presented the development, training, and evaluation of a GRU-based architecture for early stroke prediction using surface EMG signals, motivated by the global burden of stroke and the need for non-invasive, cost-effective screening.

The GRU-only model achieved a higher mean cross-validation accuracy than the CNN-GRU (55.3%) and classical Logistic Regression (47.8%) models, though the difference was not statistically significant in a paired comparison ($p = 0.125$; Section 5.1), suggesting recurrent architectures suit EMG analysis but require a larger, better-balanced dataset to confirm.

Gradient clipping and learning-rate scheduling (Section 3.5) supported stable training convergence — an actual run (Section 4.3.4) confirms smooth loss reduction and frequent clipping (≈55% of steps), with mild overfitting after epoch 4. Integrated Gradients analysis showed attribution concentrated on Channels 4-5 in the later portion of the analysis window (Figures 13-14); averaging across the full test set is needed before generalizing.

The study also revealed challenges from dataset limitations, class imbalance, and cross-validation design: the final model's evaluation split contained no stroke-labeled examples, making its 0% sensitivity a mathematical artifact rather than a measured failure (Section 5.1). A supplementary class-balanced re-evaluation (Section 4.3.3) confirms genuine, better-than-chance discrimination (ROC-AUC = 0.839, PR-AUC = 0.752) and modest calibration (Brier = 0.179), though at the cost of subject independence. These findings underscore the need for

larger, more diverse datasets and a corrected, group-independent yet class-balanced cross-validation design.

Future work should expand the dataset with more stroke patients of varying type, severity, and time since onset; implement more advanced architectures (bidirectional LSTM, attention); explore data augmentation and more effective imbalance-handling; enhance interpretability building on the Integrated Gradients results (Section 4.3.5); and design a cross-validation scheme that is both class-balanced and group-/subject-independent (Section 5.5).

Despite these limitations, the results offer a preliminary feasibility signal for GRU-based, non-invasive stroke screening from sEMG — sufficient to motivate further development but not yet clinical deployment. A larger, more diverse cohort, corrected cross-validation design, and prospective clinical validation could ultimately support earlier, lower-cost stroke detection.

Abbreviations

GRU	Gated Recurrent Unit
EMG	Electromyography
sEMG	Surface Electromyography
CNN	Convolutional Neural Network
LSTM	Long Short-Term Memory
BLSTM	Bidirectional Long Short-Term Memory
RNN	Recurrent Neural Network
ROC-AUC	Area Under the Receiver Operating Characteristic Curve
PR-AUC	Area Under the Precision-Recall Curve
DOF	Degree(s) of Freedom
AI	Artificial Intelligence
ML	Machine Learning
EEG	Electroencephalography
LR	Logistic Regression
RF	Random Forest
ANFIS	Adaptive Neuro-Fuzzy Inference System
MRI	Magnetic Resonance Imaging
EHR	Electronic Health Record
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
SD	Standard Deviation
RMS	Root Mean Square
MAV	Mean Absolute Value
WL	Waveform Length
ZC	Zero Crossings
SSC	Slope Sign Changes
IG	Integrated Gradients
SMOTE	Synthetic Minority Oversampling Technique
DALYs	Disability-Adjusted Life Years
CPU	Central Processing Unit
GPU	Graphics Processing Unit

Author Contributions

Bob Chile-Agada: Conceptualization, Formal Analysis, Methodology, Software, Writing - original draft

Laud Charles Ochei: Supervision, Validation, Writing - review & editing

Fubara Egbono: Resources, Writing - review & editing

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Adam, S. Y., Yousif, A., & Bashir, M. B. (2016). Classification of ischemic stroke using machine learning algorithms. International Journal of Computer Application, 149(10), 26-31. https://doi.org/10.5120/ijca2016911607

[2] Alhakeem, A., Chaurasia, B., & Khan, M. M. (2025). Revolutionizing stroke prediction: A systematic review of AI-powered wearable technologies for early detection of stroke. Neurosurgical Review, 48(1), 458. https://doi.org/10.1007/s10143-025-03629-4

[3] Aviles, M., Alvarez-Alvarado, J. M., Robles-Ocampo, J.-B., Sevilla-Camacho, P. Y., & Rodr ́guez-Res ́ndiz, J. (2024). Optimizing RNNs for EMG signal classification: A novel strategy using Grey Wolf Optimization. Bioengineering, 11(1), 77. https://doi.org/10.3390/bioengineering11010077

[4] Bass, L., Clements, P., & Kazman, R. (2012). Software architecture in practice (3rd ed.). Addison-Wesley.

[5] Cahuantzi, R., Chen, X., & Güttel, S. (2021). A comparison of LSTM and GRU networks for learning symbolic sequences https://doi.org/10.48550/arXiv.2107.02248

[6] Çelik, Y., & Can, Ü. (2026). Surface EMG-based hand gesture recognition using a hybrid multistream deep learning architecture. Sensors, 26(7), 2281. https://doi.org/10.3390/s26072281

[7] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., & Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078. https://doi.org/10.3115/v1/D14-1179

[8] Choi, Y.-A., Park, S.-J., Jun, J.-A., Pyo, C.-S., Cho, K.-H., Lee, H.-S., & Yu, J.-H. (2021). Deep learning-based stroke disease prediction system using real-time bio signals. Sensors, 21(13), 4269. https://doi.org/10.3390/s21134269

[9] Cuocolo, R., Cipullo, M. B., Stanzione, A., et al. (2019). Machine learning applications in prostate cancer magnetic resonance imaging. European Radiology Experimental, 3(1), 35. https://doi.org/10.1186/s41747-019-0109-2

[10] Guntari, E. W., Djamal, E. C., Nugraha, F., & Liem, S. L. L. (2020). Classification of post-stroke EEG signal using genetic algorithm and recurrent neural networks. Proceedings of the 2020 7th International Conference on Electrical Engineering, Computer Sciences and Informatics, 156-161.

[11] Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735-1780. https://doi.org/10.1162/neco.1997.9.8.1735

[12] Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. Proceedings of the 32nd International Conference on Machine Learning (ICML 2015), 448-456. https://doi.org/10.48550/arXiv.1502.03167

[13] Jha, A., Aicher, J. K., Gazzara, M. R., Singh, D., & Barash, Y. (2020). Enhanced Integrated Gradients: Improving interpretability of deep learning models using splicing codes as a case study. Genome Biology, 21, 149. https://doi.org/10.1186/s13059-020-02055-7

[14] Joloudari, J. H., Marefat, A., Nematollahi, M. A., Oyelere, S. S., & Hussain, S. (2023). Effective class-imbalance learning based on SMOTE and convolutional neural networks. Applied Sciences, 13(6), 4006. https://doi.org/10.3390/app13064006

[15] Kingma, D. P., & Ba, J. (2015). Adam: A method for stochastic optimization. Proceedings of the 3rd International Conference on Learning Representations (ICLR 2015). https://doi.org/10.48550/arXiv.1412.6980

[16] Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., & Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. Computational and Structural Biotechnology Journal, 13, 8-17. https://doi.org/10.1016/j.csbj.2014.11.005

[17] Lakkshmanan, A., Suji, A., Priyanka, N., Anand, D. B., Nagaraju, K., & Naidu, U. U. (2023). A novel machine learning based stroke prediction system using magnetic resonance image and adaptive new fuzzy inference. Intelligent Systems and Applications in Engineering, 12(10), 576-585.

[18] Loshchilov, I., & Hutter, F. (2019). Decoupled weight decay regularization. Proceedings of the 7th International Conference on Learning Representations (ICLR 2019). https://doi.org/10.48550/arXiv.1711.05101

[19] Marwat, M. (2023). MUSED-I: Surface Electromyography (sEMG) dataset [Data set]. Kaggle. https://www.kaggle.com/datasets/mustafamarwat/mused-i-semg-dataset-for-stroke-rehabilitation

[20] Schuster, M., & Paliwal, K. K. (1997). Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing, 45(11), 2673-2681. https://doi.org/10.1109/78.650093

[21] Shinde, A., Shete, V., & Mehendale, N. (2026). Optimized signal acquisition and advanced AI for robust 1D EMG classification: A comparative study of machine learning, deep learning, and reinforcement learning. Bioengineering, 13(4), 463. https://doi.org/10.3390/bioengineering13040463

[22] World Health Organization. (2019). Global Health Estimates: Life expectancy and leading causes of death and disability. Geneva: WHO Press.

[23] Xia, X., Yue, W., Chao, B., Li, M., Cao, L., Wang, L., Shen, Y., & Li, X. (2019). Prevalence and risk factors of stroke in the elderly in Northern China: Data from the National Stroke Screening Survey. Journal of Neurology, 266, 1449-1458. https://doi.org/10.1007/s00415-019-09281-5

[24] Yu, J., Park, S., Kwon, S. H., Ho, C. M. B., Pyo, C. S., & Lee, H. (2020). AI-based stroke disease prediction system using real-time electromyography signals. Applied Sciences, 10, 6791. https://doi.org/10.3390/app10196791

[25] Yuan, J., Wu, F., & Wu, H. (2023). Multivariate time-series classification using memory and attention for long and short-term dependence. Applied Intelligence, 53(24), 29677-29692. https://doi.org/10.1007/s10489-023-05079-1