**Explainable State-of-Health Estimation for Lithium‑Ion Batteries**

Author: Baze Bai

**Research Question & Hypotheses**  
Estimating the State‑of‑Health (SOH) of lithium‑ion batteries accurately remains challenging, and few studies examine model explainability. This project tests two hypotheses:

1. No specific patterns exist between external features and SOH.
2. Distinct patterns do exist between external features and SOH. ​

**Importance**  
SOH monitoring is vital for battery safety, longevity, and cost‑effective maintenance. Adding explainability helps reveal which external measurements truly drive SOH predictions, leading to more reliable management. ​

**Prior Work**  
Wang et al. (“Explainability‑driven model improvement for SOH estimation of lithium‑ion Battery”) combined deep learning (CNN/LSTM) with post‑hoc methods like LRP to identify key features and feed those insights back into training, improving both interpretability and accuracy. ​

**Proposed Approach**  
• Build a multi‑channel 1D‑CNN where each channel processes a different feature set (e.g., EIS, IC curves, temperature).  
• Apply Integrated Gradients (IG) maps across SOH levels (100%→70%) to visualize and quantify feature importance. ​

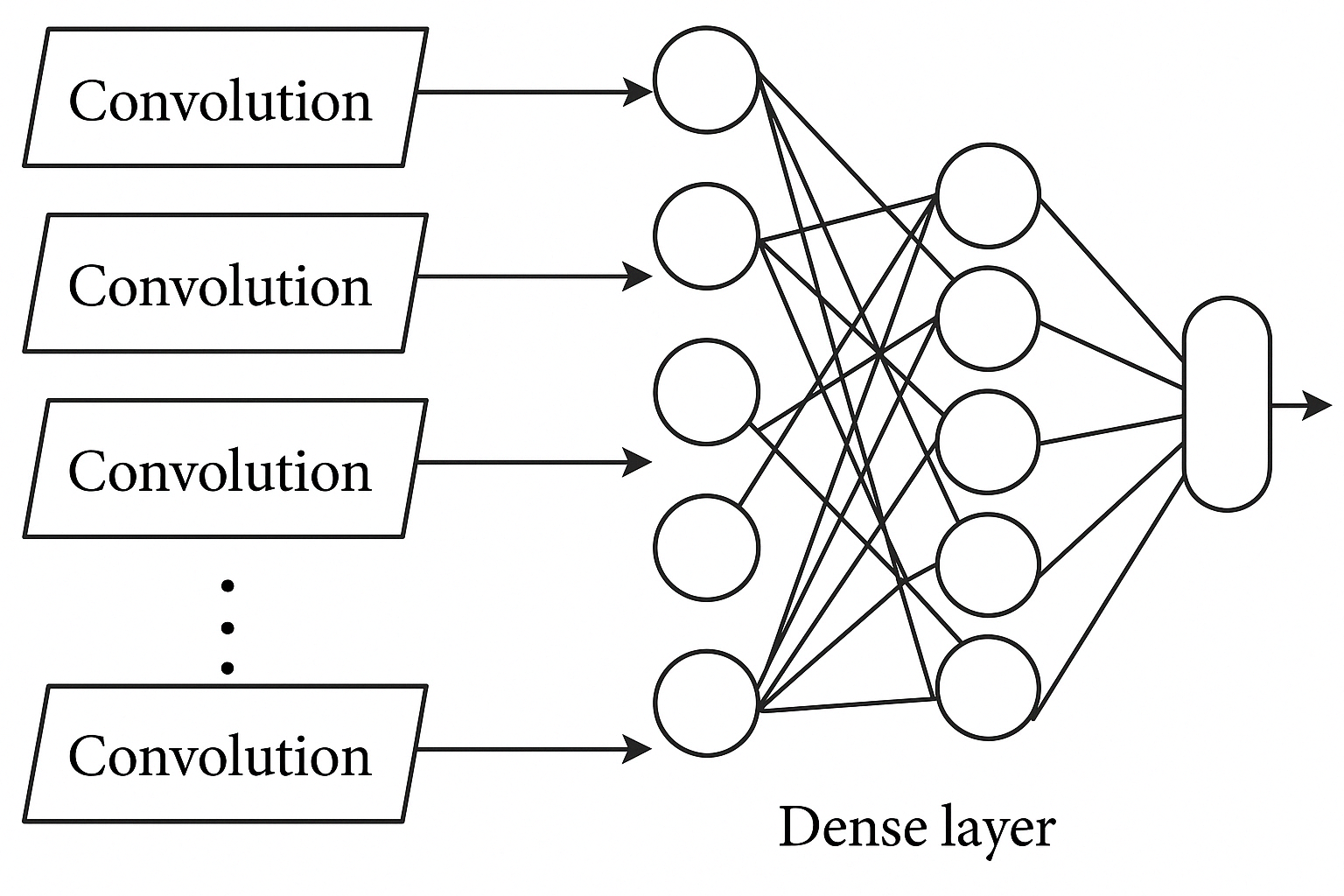


Fig.1 Architecture of 1D-CNN

## **MultiSource1DCNN Architecture**

**Input**: (Batch, n sources, feature\_length)

**Processing**:

* n parallel convolutional branches (one per source)
* Each branch: 3 Conv1D layers (1→16→32→64 channels) with ReLU
* All convolutions maintain sequence length (60)

**Feature Fusion**: Concatenate outputs from all branches

**Classification**:

* Flatten concatenated features
* Two FC layers (flattened→128→num\_classes)
* Default output: Single value for regression/binary classification

The model processes multiple time series independently before combining features for prediction.

**Explain model with Integrated Gradients**

1. It takes a trained model, input data, optional target class indices, baseline inputs (defaults to zero tensor), and number of interpolation steps.
2. The function uses Captum's Integrated Gradients to calculate attribution scores by:

* Creating a path from baseline to input
* Computing gradients at each step along this path
* Averaging these gradients and multiplying by input-baseline difference

1. This reveals which input features (time points and variables) most influenced the model's predictions, with higher attribution values indicating greater importance.
2. The attribution tensor (shape: batch × variables × time) quantifies each input feature's contribution, helping understand model behavior by highlighting which parts of the input most affect the output.
3. The function also returns convergence deltas to verify the approximation quality of the numerical integration.

Result:

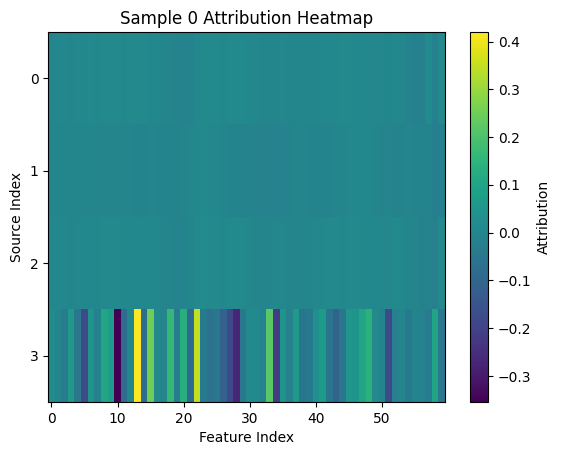


Fig.2 Integrated Gradients Map

This heatmap visualizes feature attributions for a single sample (Sample 0). The x-axis represents different feature indices, while the y-axis shows source indices (e.g., channels or input segments). Bright colors (yellow) indicate high positive attribution, dark colors (purple) represent negative attribution, and teal indicates near-zero contribution. Notably, only source index 3 shows significant attribution activity, suggesting it is the most influential input source for the model’s output in this sample.