

String Health Algorithm

Tự Huy

December 2025

1 Chiến lược LSTM Autoencoder cho Phát hiện Bất thường

Dựa trên bộ dữ liệu hiện tại (`string_health_5min.csv`, đã gồm bức xạ và nhiệt độ giả lập) và phần xử lý trong notebook trước đó, em đề xuất sử dụng chiến lược **LSTM Autoencoder (Reconstruction-based)**. Phương pháp này khắc phục hạn chế của phiên bản trước bằng cách bổ sung các biến ngoại sinh (G, T), giúp mô hình hiểu rõ hơn hành vi vật lý của tấm pin.

1.1 Triết lý cốt lõi: Tái tạo thay vì Dự báo

Thay vì dự đoán sản lượng tương lai, mô hình được huấn luyện để trả lời câu hỏi: “*Với bức xạ G và nhiệt độ T tại thời điểm hiện tại, dòng điện I và điện áp V lẽ ra phải có giá trị bao nhiêu?*”

- Khi hệ thống bình thường: giá trị thực tế \approx giá trị tái tạo.
- Khi có lỗi (Imbalance hoặc Disconnection): giá trị thực tế \ll giá trị tái tạo.
- Sai số này được gọi là **Reconstruction Error (RE)**, và là cơ sở để đặt ngưỡng phát hiện lỗi.

1.2 Chiến lược phương pháp (Methodology)

Bước 1: Feature Selection & Scaling

Trong phiên bản cũ, mô hình chỉ dùng V, I, P , nên không học được tác động của bức xạ. Với dataset mới, vector đầu vào tại thời điểm t được thiết kế như sau:

$$X_t = [G, T_{\text{panel}}, V_{dc}, I_{dc}, P_{dc}]$$

Tất cả được chuẩn hoá về khoảng $[0, 1]$ bằng `MinMaxScaler`.

Bước 2: Sliding Window

Mô hình quan sát một đoạn lịch sử để nắm bắt ngữ cảnh.

- Độ dài chuỗi: $L = 12$ bước (60 phút).
- Input shape: $(\text{Batch_Size}, 12, 5)$.

Bước 3: Kiến trúc LSTM Autoencoder

Mạng gồm hai phần:

1. **Encoder**: LSTM nén chuỗi 12 bước thành vector tiềm ẩn.
2. **Decoder**: Tái tạo lại chuỗi ban đầu từ vector này.

Hàm mất mát sử dụng:

$$\text{Loss} = \text{MAE}(X_{\text{input}}, X_{\text{output}})$$

Mô hình học tốt dữ liệu bình thường, nhưng không thể tái tạo chính xác dữ liệu lỗi
→ sai số tăng mạnh.

Bước 4: Huấn luyện trên dữ liệu sạch

- Mô hình được train trên tập Train, giả định đa số là bình thường.
- Khuyến nghị chỉ dùng dữ liệu ban ngày: $G > 50$.

Bước 5: Tính toán Reconstruction Error

Sai số tại thời điểm t được tính như sau (tập trung vào dòng điện I):

$$RE_t = |I_{\text{thực}} - I_{\text{tái tạo}}|$$

1.3 Xác định Ngưỡng (Threshold)

Vì không có nhãn, ngưỡng được xác định bằng thống kê trên tập Train. Giả định rằng các điểm sai số lớn nhất trong Train chỉ là nhiễu.

Cách 1: Quy tắc 3-Sigma

$$\text{Threshold} = \mu_{RE} + 3\sigma_{RE}$$

Cách 2: Phân vị (Percentile) – Khuyến nghị

$$\text{Threshold} = \text{Percentile}(RE_{\text{train}}, 99.5)$$

1.4 Output cuối cùng

Ngưỡng có thể được tính bằng đoạn mã sau:

```
threshold = np.percentile(train_mae_loss, 99.9)
# hoặc:
# threshold = np.max(train_mae_loss)
```

Quy tắc phát hiện lỗi:

$$|I_{\text{thực}} - I_{\text{tái tạo}}| > \textit{Threshold} \Rightarrow \text{BÁO LỖI (1)}$$

Ngược lại:

BÌNH THƯỜNG (0)

Phương pháp này tuân theo triết lý **Zero-Positive**: ưu tiên độ chắc chắn cao khi báo lỗi, chấp nhận bỏ sót lỗi nhỏ nhưng đảm bảo rằng mỗi cảnh báo đều có ý nghĩa thực sự.