NILM Model Comparison

LSTM vs Random Forest vs SVM

Student: D11351004 陳重光 (Alan Chen)

Final Presentation

Motivation & Goal

Why Non-Intrusive Load Monitoring?

Low-Cost Monitoring: Enables energy tracking using only aggregate meter data, avoiding expensive sub-metering hardware.

Practical Insights: Real-world datasets allow for effective appliance disaggregation and usage pattern analysis.

Research Goal: Benchmark **Random Forest (RF)** against standard **Neural Network (LSTM)** models to identify the optimal approach for household energy data.

Dataset Overview

Self-collected weekly household power consumption (example day shown)

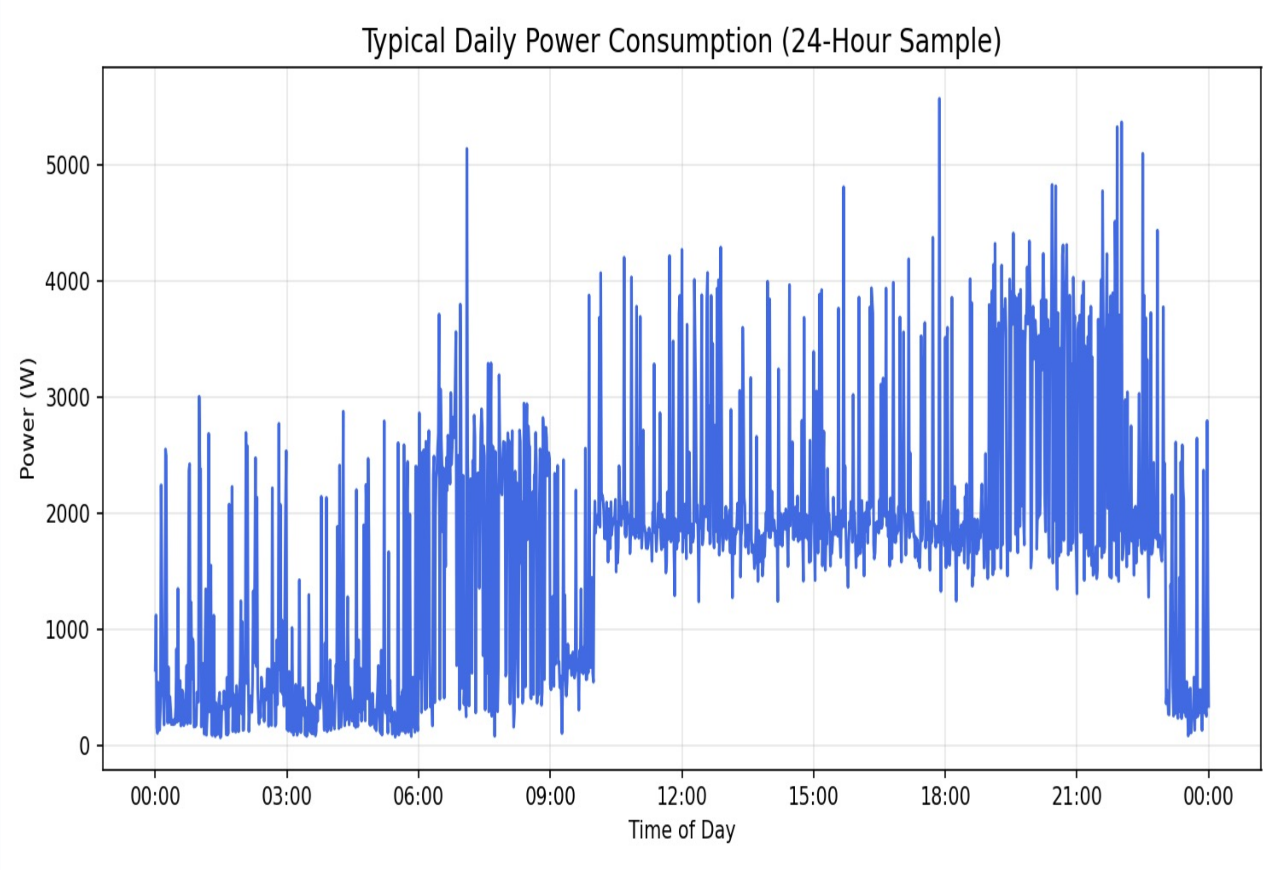
Key Characteristics

Synthetic (simulated) aggregate power signal generated for rapid ML pipeline testing.

Includes realistic-looking appliance cycles and stochastic noise to mimic fluctuations.

Labels are rule-based: system_on / power_state are defined by thresholds on the aggregate power (used as ground truth for this demo).

This plot shows a representative 24-hour simulated sample for visualization; experiments use the full synthetic dataset.



Typical Daily Power Consumption (24-hour sample)

Models Compared

Deep temporal vs feature-based vs kernel classifier

LSTM

Learns temporal patterns directly from sequences.

Good for raw signal modeling (minimal feature engineering).

Higher compute cost; needs careful regularization.

Random Forest

Ensemble of decision trees (feature-based).

Handles non-linearity and small datasets well.

Provides feature importance; fast inference.

SVM

Margin-based classifier (linear / kernel).

Effective with limited samples and good features.

Sensitive to scaling and hyperparameters.

Implementation Pipeline

From raw power signal to evaluation

Preprocess
Clean + normalize
handle missing

→

Windowing
Fixed-length
segments

→

Feature / Sequence
RF/SVM: features
LSTM: sequences

→

Train
Tune hyperparams
(CV / val)

→

Evaluate
CM, Acc,
Macro F1

Notes

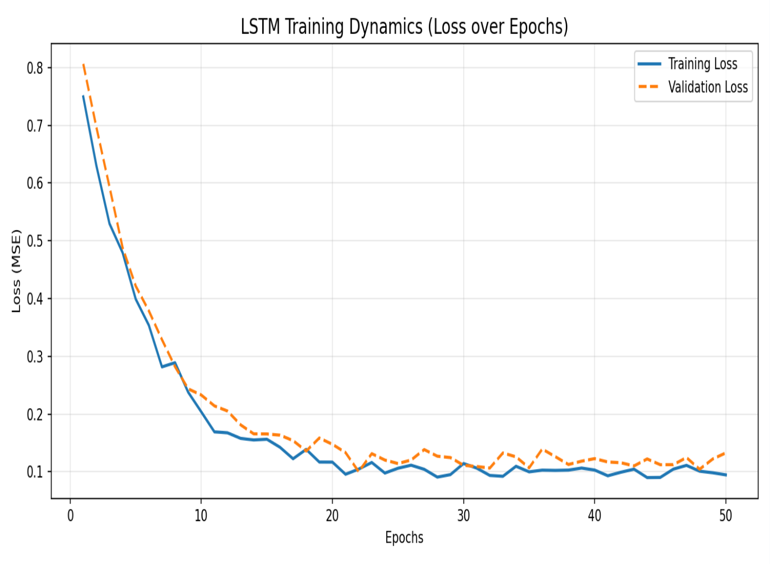
RF/SVM require engineered features; LSTM consumes sequences directly.

Keep the same train/test split across models for fair comparison.

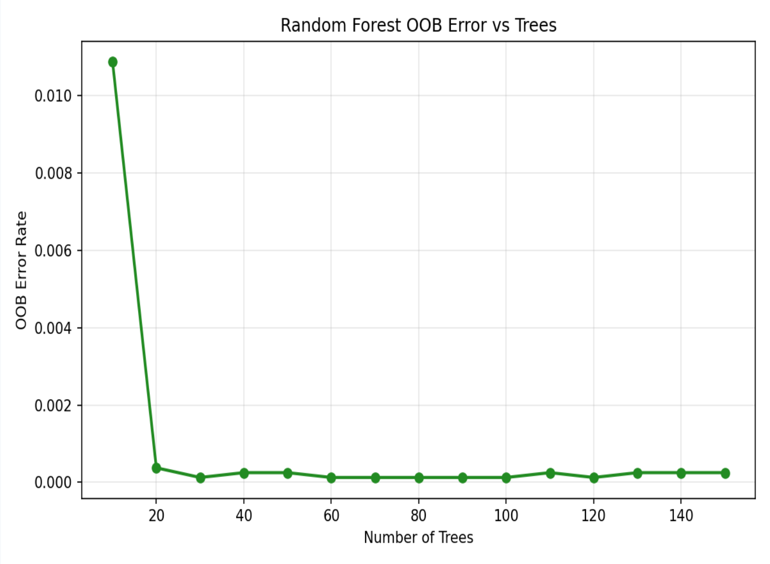
If using sliding windows, ensure no overlap across train/test to avoid leakage.

Training Dynamics

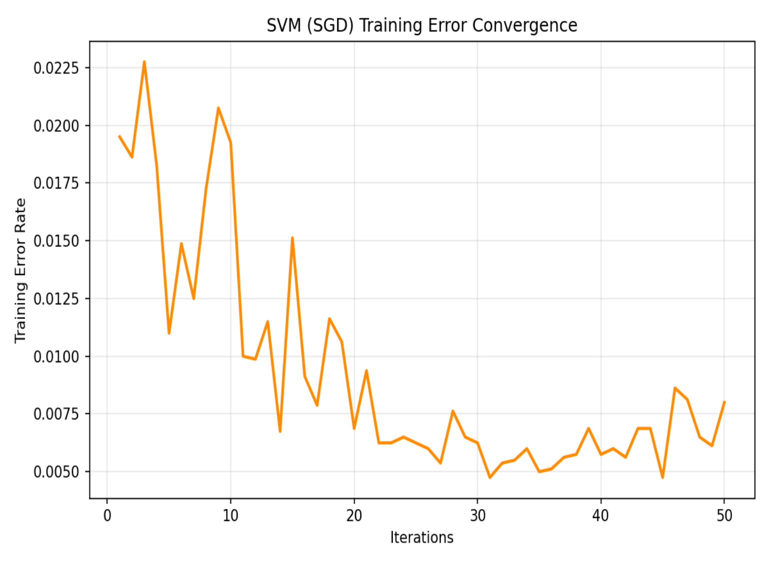
Learning curves / convergence behaviors



LSTM (Loss)



RF (OOB Error)

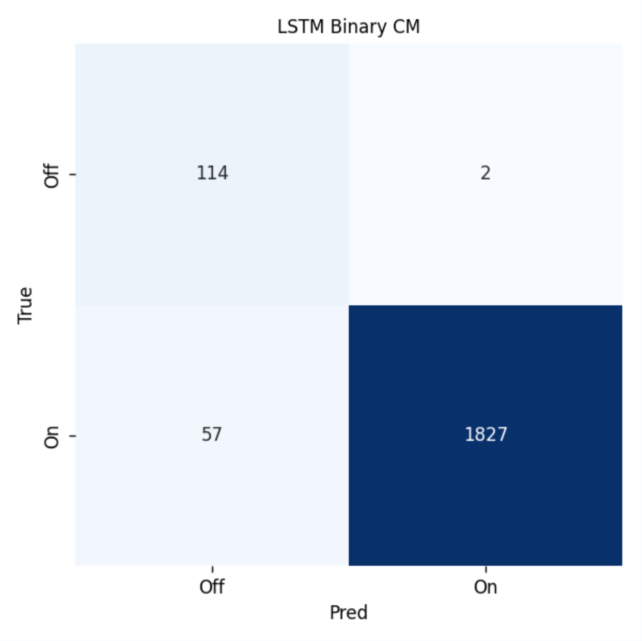


SVM (SGD) (Error)

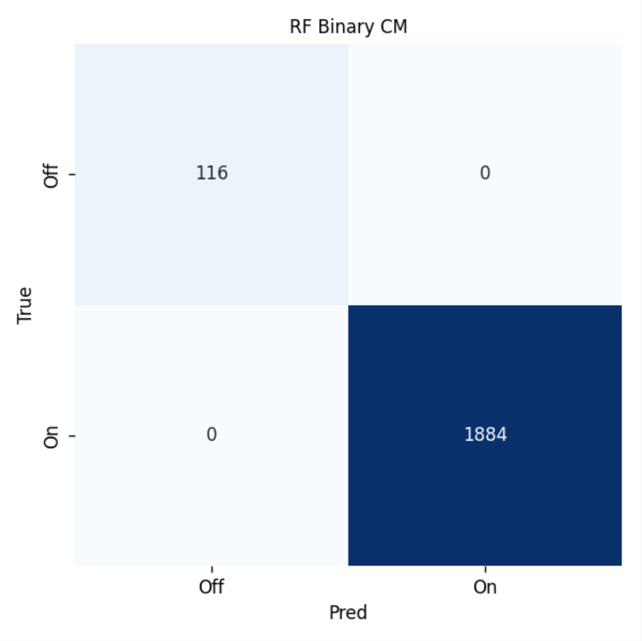
Observation:
RF quickly reaches near-zero OOB error;
LSTM converges smoothly but requires epochs;
SVM converges with some oscillation.

Binary Classification Results (ON/OFF)

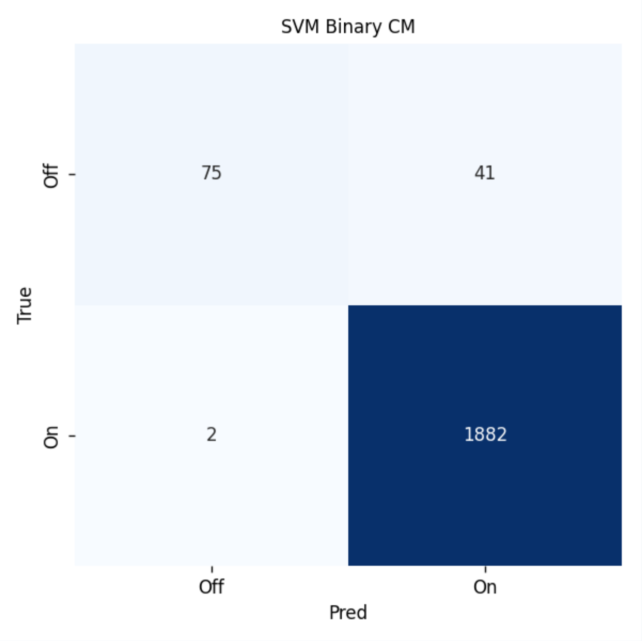
Confusion matrices + corrected metrics



LSTM
Acc 97.05% | Macro F1 88.93%



Random Forest
Acc 100.00% | Macro F1 100.00%



SVM
Acc 97.85% | Macro F1 88.30%

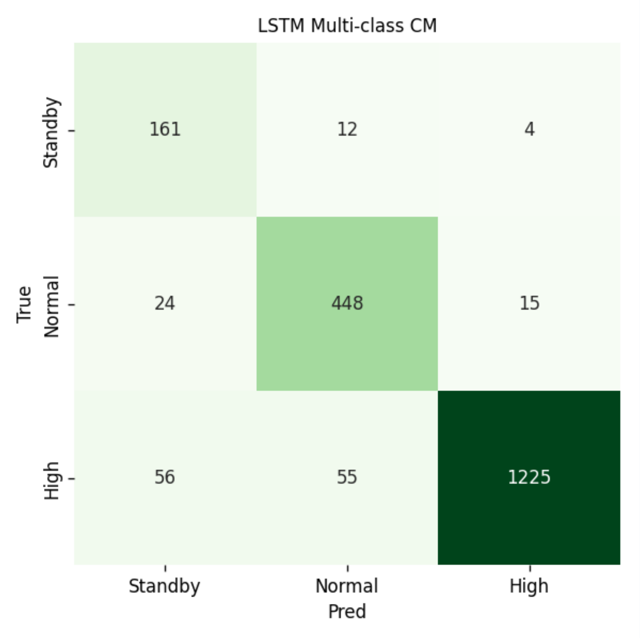
Findings:

RF achieves perfect separation on this test set (2000 samples).

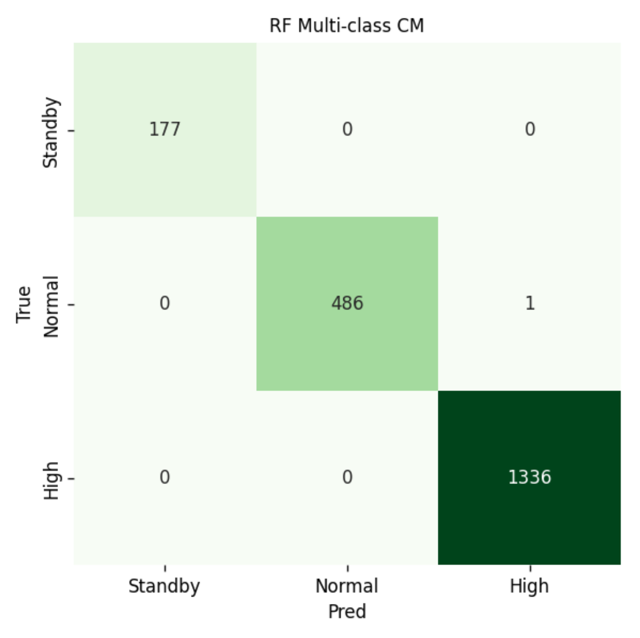
LSTM and SVM are strong; most errors are ON→OFF or OFF→ON confusions under overlap/noise.

Multi-class Results (3-State)

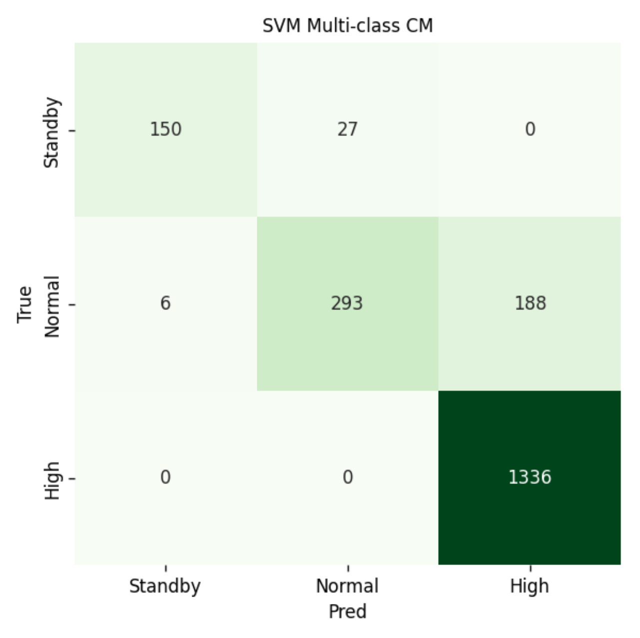
Standby / Normal / High



LSTM
Acc 91.70% | Macro F1 87.14%



Random Forest
Acc 99.95% | Macro F1 99.95%



SVM
Acc 88.95% | Macro F1 85.38%

Findings:

Main difficulty is separating 'Normal' vs 'High' for LSTM/SVM; RF stays almost perfect.

Report both Accuracy and Macro F1 to reflect class imbalance.

Performance Summary

Metrics derived from confusion matrices (test set size = 2000)

Model	Binary Acc	Binary Macro F1	3-State Acc	3-State Macro F1
LSTM	97.05%	88.93%	91.70%	87.14%
RF	100.00%	100.00%	99.95%	99.95%
SVM	97.85%	88.30%	88.95%	85.38%

Key Insights

Random Forest dominates on this dataset, likely due to distinct statistical patterns and clean labeling.

LSTM is competitive for sequence modeling but may need more data / augmentation for multi-state separation.

SVM depends heavily on feature scaling and choice of kernel / regularization.

Conclusion & Future Work

What to improve next

Conclusion

On this dataset, Random Forest is the best performer for both binary and 3-state classification.

LSTM remains a strong choice when using raw sequences, especially as data scale and noise increase.

SVM is a solid baseline but sensitive to features and hyperparameters.

Future Work (to strengthen correctness/generalization)

Since labels are rule-based in this demo, future work will focus on appliance-level ground truth and more realistic data generation. Use time-aware splits (e.g., train on earlier days, test on later days) to avoid leakage from overlapping windows.

Evaluate on noisier / multi-appliance scenarios; report per-class precision/recall/F1.

Try feature ablation for RF and data augmentation / class weighting for LSTM.