

Phase Plan

Đề tài. Phân loại tình trạng ổ bi từ tín hiệu rung (CWRU làm bộ dữ liệu lỗi).

Bài báo. *Vibration signal analysis for rolling bearings faults diagnosis based on deep-shallow features fusion.*

Tài liệu môn học. Bishop — *Pattern Recognition and Machine Learning* (2006).

Phương pháp. Chuỗi xử lý tham chiếu: log-mel \rightarrow MBH-LPQ + VGGish \rightarrow PCA \rightarrow EDA \rightarrow so khớp cosine \rightarrow tổng trọng số điểm (WS).

Mô hình. Nhánh đặc trưng nông (LBP, LDP, LPQ, MBH-LPQ) và nhánh sâu (VGGish, VGG16, YamNet); đánh giá *fusion* ở mức đặc trưng và mức điểm.

Nguyên tắc chung. Mỗi phase có đầu ra rõ ràng, tham số lưu bằng tệp cấu hình, seed cố định; tách bạch train/val/test; kết quả có thể tái lập.

Phase 0 — Xác lập đề tài và phạm vi

Mục tiêu:

Chốt tên đề tài, mục tiêu, phạm vi ứng dụng.

Nội dung: • Chuẩn hoá thuật ngữ và phạm vi đánh giá.

- Tài liệu hoá mục tiêu và kỳ vọng kết quả.

Kết quả:

Tệp mô tả phạm vi và mục tiêu nằm trong [docs/requirements.md](#).

Phase 1 — Khung dự án, môi trường và tái lập

Mục tiêu:

Tạo khung thư mục, môi trường conda, quy trình tái lập.

Nội dung: • Cấu trúc: configs/, data/, docs/, experiments/, notebooks/, original_paper/, results/, src/, tests/.

- environment.yml cho môi trường ndm; cố định seed trong cấu hình.
- .gitignore: bỏ qua data/raw/CWRU/**, results/**, *.zip.
- README có mục **Download (Release)** dùng liên kết /releases/latest/download/... và huy hiệu “Release”.

Kết quả:

Dự án *clean* có thể cài đặt và chạy kiểm thử; [docs/reproducibility.md](#).

Phase 2 — Đọc bài báo và đối sánh nội dung học phần

Mục tiêu:

Nắm rõ pipeline tham chiếu và liên hệ học phần.

Nội dung:

- Tóm tắt bài báo, trích các giả thiết và tham số chính.
- Bảng ánh xạ khối xử lý \leftrightarrow nội dung môn học.

Kết quả:

`papers/sciRep2025_summary.md`, `docs/course_mapping.md`.

Phase 3 — Dữ liệu: tải, mô tả, hệ thống nhãn

Mục tiêu:

Tổ chức dữ liệu CWRU đầy đủ và nhất quán.

Nội dung:

- Tải về `data/raw/CWRU/`: `normal_baseline/Normal_0..3.mat`.
- Thư mục `48k_drive_end_fault/` với các lỗi: B007, B014, B021, IR007, IR014, IR021, OR007@3/6/12, OR014@6, OR021@3/6/12; đủ các tải `_0`, `_1`, `_2`, `_3`.
- Chuẩn hoá tên lớp, sinh `data/labels.csv` và `data/splits/cwru_splits.json`.

Kết quả:

`docs/data_description.md`, `labels.csv`, `splits/cwru_splits.json`.

Phase 4 — Nạp dữ liệu, tách tập, cắt đoạn/chuẩn hoá

Mục tiêu:

Xây dựng mô-đun nạp dữ liệu tin cậy, không rò rỉ.

Nội dung:

- Tách theo *file* trước khi cắt đoạn; chuẩn hoá biên độ.
- Sõ tay khảo sát phân phối; tham số hoá qua `configs/*.yaml`.

Kết quả:

`src/dataio.py`, `notebooks/01_explore_data.ipynb`.

Phase 5 — Tuyển cơ sở ML theo PRML

Mục tiêu:

Thiết lập mốc so sánh cổ điển.

Nội dung:

- Đặc trưng thời gian/tần số; Logistic/Softmax, SVM, Random Forest.
- Chọn tham số trên *validation*, đánh giá trên *test*.

Kết quả:

`results/baseline_prml_metrics.csv`, `results/cm_prml.png`.

Phase 6 — Spectrogram chuẩn cho CNN và pipeline bài báo

Mục tiêu:

Chuẩn hoá biến đổi thời-tần cho nhánh sâu và log-mel.

- Nội dung:**
- STFT \rightarrow log-magnitude hoặc log-mel; cố định cửa sổ, bước trượt, `nfft`, số dải mel.
 - Notebook xem trước để kiểm tra trực quan.

Kết quả:

`notebooks/04_spectrogram_preview.ipynb` và ảnh minh hoạ.

Phase 7 — CNN cơ sở và trích đặc trưng sâu

Mục tiêu:

Có mốc sâu cơ sở và embedding để kết hợp.

- Nội dung:**
- Huấn luyện CNN trên spectrogram; xuất embedding lớp gần cuối.

Kết quả:

`results/cnn_baseline.pt`, `results/cm_cnn.png`, `X_deep.npy`.

Phase 8 — Kết hợp đặc trưng và điểm số

Mục tiêu:

Đánh giá kết hợp mức đặc trưng và mức điểm, chọn cấu hình hiệu quả.

- Nội dung:**
- Nối sâu-nông và huấn luyện bộ phân lớp.
 - Kết hợp điểm kiểu WS; quét hệ số trên *validation*, cố định cho *test*.

Kết quả:

`results/fusion_feature_metrics.csv`, `results/fusion_score_metrics.csv`.

Phase 9 — *Paper-core* (tái hiện nghiêm ngặt)

Mục tiêu:

Tái hiện pipeline của bài báo trên CWRU với giao thức thống nhất.

- Nội dung:**
- Giao thức: `fs=48k`, `rpm=1772`, `load=1HP`, 10 lớp; chia train/test 80/20; tách train thành train/val.
 - Mỗi waveform \rightarrow 100 mẫu; mỗi mẫu 4710 điểm; STFT Hamming, overlap 50%, `nfft=512`.
 - Log-mel cho VGGish (96 \times 64, 64 mel bins); LPQ với $R \in \{3, 5, 7\}$; MBH-LPQ số khối $b \in \{1, 2, 4, 6, 8, 10, 12\}$.
 - WS quét $\alpha \in \{0.1, \dots, 0.9\}$; báo cáo các mốc shallow và deep (YamNet, VGG16, VGGish).

Kết quả:

`original_paper/method_sciRep_like.py`, `original_paper/00_run_sciRep_like.ipynb`,

results/paper_core_metrics.csv, results/cm_paper_core.png,
results/b_sweep.csv + hình, results/ws_sweep.csv + hình.

Phase 10 — Mở rộng: PU hoặc *domain shift* trên CWRU

Mục tiêu:

Đánh giá khả năng khái quát.

Nội dung: • Chạy lại pipeline *paper-core* trên PU (nếu sẵn có) hoặc kịch bản train load A, test load B trên CWRU.

Kết quả:

Bảng so sánh và nhận xét tổng hợp.

Phase 11 — Ablation và kiểm thử nhiều

Mục tiêu:

Hiểu vai trò các thành phần và độ bền với nhiễu.

Nội dung: • So sánh: thời vs tần; sâu vs nông vs kết hợp; thay đổi cửa sổ.
• *Domain shift*; nhiễu Gaussian SNR $\in \{-6, -3, 0, 3, 6\}$ dB.

Kết quả:

results/ablation_summary.csv, results/ablation_plots.png, docs/analysis_notes.md.

Phase 12 — Tự động hoá chạy và phát hành Release

Mục tiêu:

Một lệnh tái tạo toàn bộ và công bố gói kết quả.

Nội dung: • `python -m experiments.run_all -config configs/exp_*.yaml`.
• Sinh results/manifest.json (commit, seed, phiên bản thư viện, danh sách artefact).
• GitHub Release vX.Y.Z đính kèm NDM_Project_full.zip, CWRU_dataset.zip, results_artifacts.zip.
• README dùng /releases/latest/download/...; kèm hướng dẫn kiểm tra sha256.

Kết quả:

Bộ artefact và trang phát hành vX.Y.Z.

Phase 13 — Ghi chú kỹ thuật và công thức

Mục tiêu:

Chuẩn hoá nền tảng kỹ thuật dùng trong đề án.

Nội dung: • STFT/mel, LPQ/MBH-LPQ, PCA/EDA, cosine, WS, SNR; định nghĩa, giả thiết, công thức.

Kết quả:

docs/math_notes.md.

Phase 14 — Báo cáo và trình bày

Mục tiêu:

Hoàn thiện hồ sơ học thuật của đề án.

Nội dung: • Push Git, soạn báo cáo và slide; kiểm tra liên kết, số liệu, hình ảnh; đối chiếu với manifest.

Kết quả:

report.pdf, slides.pdf, thẻ v1.0.