# Nhận diện Âm thanh từ Dữ liệu BirdCLEF+ 2025

*Project Deep-learning

1st Lê Anh Tiến
*22022528*

2nd Vũ Văn Phong
*22028309*

***Tóm tắt nội dung*—trong báo cáo này đề cập đến giải pháp để phân loại tiếng chim thông qua 2 mạng học sâu nơ ron tích chập CNN và model efficientNet. Báo cáo đề cập đến phần xử lý dữ liệu , training mô hình, hiệu năng,...**

***Index Terms*—BirdCLEF 2025, bioacoustic classification, CNN, EfficientNet, soundscape analysis, spectrogram, deep learning.**

## I. Giới thiệu

Việc phân loại tiếng chim giúp hỗ trợ các nhà khoa học theo dõi và phát hiện sớm sự tuyệt chủng của nó. Tuy nhiên tiếng trong môi trường bị nhiễu bởi tiếng mưa, côn trùng,... Điều này đặt ra yêu cầu rất lớn về độ mạnh của những mô hình học sâu để học được đặc trưng âm thanh và phải có khả năng khái quát hóa tốt trên dữ liệu không đồng nhất.

## II. Dữ liệu

### A. Mô tả dữ liệu

Tập dữ liệu BirdCLEF+ 2025 gồm:

- **train_audio**: các file âm thanh ngắn (1–10 s, .ogg, 32 kHz) đã gán nhãn loài.
- **train_soundscapes**: các đoạn âm thanh dài (10–15 phút) chưa gán nhãn.
- **test_soundscapes**: các file 1 phút (ogg, 32 kHz), tên dạng `soundscape_xxxxxx`.ogg.
- **train.csv**: metadata cho `train_audio`, bao gồm `filename`, `primary_label`, `secondary_labels`, `latitude`, `longitude`, `author`, `rating`, `collection`.
- **taxonomy.csv**: thông tin phân loại sinh học (mã loài, lớp, họ, chi).
- **recording_location.txt**: mô tả vị trí ghi âm (El Silencio Natural Reserve) và tọa độ.
- **sample_submission.csv**: mẫu nộp với `row_id` và 206 cột `species_id`.

### B. Khám phá Dữ liệu (EDA)

- **Quy mô và nhãn:**
  - Tập dữ liệu gồm hàng chục nghìn bản ghi âm, đại diện cho hơn 1.000 loài chim khác nhau.
  - **Nhãn chính (`class_name`):**
    - Tổng số loài: ∼1.000.
    - Top 5 loài phổ biến nhất (theo số bản ghi): *Silvia borin*, *Turdus merula*, *Parus major*, *Phylloscopus trochilus*, *Erithacus rubecula*.
  - **Nhãn phụ (`secondary_labels`):**
    - Khoảng 80 % bản ghi chỉ có nhãn chính, còn lại có 1–2 nhãn phụ đi kèm (ví dụ: tiếng báo động, tiếng gáy lẫn tiếng hót).
  - **Bộ sưu tập (`collection`):**
    - Bao gồm các nguồn như “expert”, “citizen science”, “test”...
    - “expert” chiếm ∼40 %, “citizen science” chiếm ∼50 %, phần còn lại là bộ test chưa gán nhãn đầy đủ.
- **Giá trị thiếu và chất lượng:**
  - Hơn 800 bản ghi thiếu vĩ độ/kinh độ, phải xử lý riêng.
  - **Rating:**
    - Thang 0–5, với phần lớn bản ghi rơi vào khoảng 3–4.
    - Ít hơn 5 % bản ghi có rating dưới 1 hoặc đúng 5.
- **Phân bố theo tọa độ (Geography):**
  - Sau khi loại bỏ bản ghi thiếu tọa độ, vẫn còn ∼20.000 điểm thu âm hợp lệ.
  - **Scatter plot kinh độ–vĩ độ:**
    - Tập trung dày đặc ở vùng nhiệt đới: Đông Nam Á, Nam Mỹ (Amazon), Trung Phi.
    - Một số điểm rải rác ở châu Âu và Bắc Mỹ, cho thấy đa dạng môi trường thu âm.
  - **Bản đồ Folium:**
    - Mỗi điểm vẽ dưới dạng vòng tròn, kích thước tỉ lệ với $\sqrt{\text{rating}}$ (thể hiện chất lượng thu âm).
    - Popup hiển thị `primary_label`, `secondary_labels`, `type` và đường dẫn URL, giúp nhanh chóng xem thông tin chi tiết mỗi bản ghi.

### C. Tiền xử lý dữ liệu

Phần này mô tả chi tiết từng bước tiền xử lý, bao gồm công thức, đơn vị và ý nghĩa khoa học:

1) **Đọc và hợp nhất metadata** *(Xác thực nhãn và chất lượng)*
   - Đọc `train.csv` để lấy `filename`, `primary_label`, `rating` (1–5).
   - Đọc `taxonomy.csv` để gán lớp sinh học `class_name` cho mỗi nhãn.
   - Kết quả: DataFrame chứa cột `filepath` và `target` (int).

- Đảm bảo mỗi mẫu âm thanh được gán đúng nhãn và có thông tin để lọc hoặc cân nhắc trọng số.

2) **Chuẩn hoá độ dài và tần số mẫu** *(Đồng nhất đầu vào)*
   - Tần số mẫu: $FS = 32{,}000$ Hz.
   - Thời lượng: $D = 5.0$ s. Số mẫu: $N = FS \times D = 160{,}000$.
   - Với file ngắn: dùng *padding* (zero-pad); file dài: *cắt giữa* để lấy $N$ mẫu.
   - Thống nhất kích thước cho batching và tránh sai khác độ dài.
3) **Chuyển thành Mel-spectrogram** *(Miền tần số theo thính giác)*
   - Áp dụng STFT:
   $$S(m,k) = \sum_{n=0}^{N-1} y[n]\, w[n-kH]\, e^{-2\pi i m n/M},$$
   - Mel filterbank:
   $$M(p,k) = \sum_{m=0}^{M-1} |S(m,k)|^2\, H_p(m),$$
   - Cấu hình: $n\_fft = 1024,\ hop\_length = 512,\ n\_mels = 128,\ f_{min} = 50,\ f_{max} = 14000.$
   - Mô phỏng cách tai người nghe, giúp mạng học đặc trưng âm sắc.
4) **Chuyển sang decibel và chuẩn hoá cường độ** *(Nén động và ổn định)*
   - Dùng:
   $$M_{dB}(p,k) = 10\log_{10}(M(p,k) + \varepsilon), \quad \varepsilon = 10^{-8}.$$
   - Chuẩn hoá:
   $$M_{norm} = \tfrac{M_{dB} - \min M_{dB}}{\max M_{dB} - \min M_{dB} + \varepsilon}.$$
   - Nén dải động, giảm nhiễu mạnh, ổn định gradient.
5) **Resize và lưu trữ** *(Chuẩn hoá hình ảnh và tiết kiệm)*
   - Dùng `cv2.resize` để đồng nhất $(256 \times 256)$.
   - Lưu Mel-spectrogram vào file `.npy` để tái sử dụng.
   - Giảm thời gian tính toán lặp lại, tối ưu I/O.
6) **Chú thích về thang Mel** *(Cân nhắc thính giác)*
   - Công thức chuyển $f$ thành Mel:
   $$\text{Mel}(f) = 2595\log_{10}(1 + \tfrac{f}{700}).$$
   - Mô phỏng phân giải tần số của tai người, ưu tiên vùng thấp.

### D. Dataset Preparation and Data Augmentations

Để chuẩn bị dữ liệu cho mô hình, chúng tôi thực hiện các bước sau:

- **Chuyển đổi thành Mel-spectrogram:** – Áp dụng STFT với $n\_fft = 1024$, $hop\_length = 512$ để tính spectrogram công suất $S(m,k)$. – Sử dụng Mel filterbank ($n\_mels = 128$, $f_{min} = 50$Hz, $f_{max} = 14000$Hz) để thu được $M(p,k)$. – Chuyển sang dB và chuẩn hóa – Kết quả: ảnh Mel-spectrogram $256 \times 256$ pixel, chuẩn cho CNN.
- **Tăng cường dữ liệu (Augmentation, 50% mỗi loại):** *Time Stretching*: giãn/nén thời gian với hệ số ngẫu nhiên [0.8,1.2], giúp mạng chịu được thay đổi tốc độ chim hót. *Pitch Shifting*: dịch cao độ ±2 semitone, cho phép học âm sắc bất chấp biến đổi tông. *Volume Adjustment*: điều chỉnh gain trong [0.8,1.2] và bias ±0.1, mô phỏng sự khác biệt cường độ. *SpecAugment*:
  - Time Masking: che 1–3 dải thời gian ngang (5–20 frames).
  - Frequency Masking: che 1–3 dải tần số dọc (5–20 bins).
- **Dataset Object (BirdCLEFDatasetFromNPY):** – Nạp sẵn Mel-spectrogram từ file `.npy` hoặc tính real-time nếu chưa có. – Ánh xạ nhãn chính và phụ thành vector multi-hot. – Chỉ áp dụng augmentation khi `mode="train"`.
- **Custom collate_fn:** Gom batch thành các tensor đồng nhất, đồng thời giữ lại danh sách `filename` để debug.
- **Lợi ích:** – Tăng tính đa dạng cho dữ liệu, giảm overfitting. – Ổn định đầu vào qua chuẩn hóa, resize. – Hỗ trợ multi-label và flexible loading.

## III. MODEL

- Hiểu rõ mục tiêu và cấu trúc tổng quan.
- Xem xét cách định nghĩa mô hình và chiến lược huấn luyện.
- Đánh giá phương pháp đánh giá (validation) và kết quả đạt được.

### A. Baseline CNN

Mô hình baseline CNN được sử dụng để huấn luyện đầu tiên

*Cách tiếp cận:*

- **Pipeline end-to-end**: âm thanh (audio) → Mel-spectrogram (hình ảnh) → CNN → xác suất → file nộp.
- **Chuyển đổi domain**: Dùng Mel-spectrogram để tận dụng mô hình xử lý ảnh CNN (ResNet).
- **Baseline dummy**: Chạy nhanh trên mẫu nhỏ (10 ảnh) để xác nhận luồng xử lý đúng.
- **Mục tiêu**: Đạt kết quả khả thi (toy example), làm nền tảng cho các mô hình phức tạp hơn.

*Chi tiết mô hình:*

a) **Chuẩn bị dữ liệu (SpectrogramDataset)**
   - Lưu Mel-spectrogram dưới dạng file PNG ($256 \times 256$ px @100 DPI).
   - Chỉ sử dụng 10 ảnh mẫu ban đầu để demo.

- Transform: resize $(224 \times 224)$, normalize, chuyển thành tensor 3 kênh.

b) **Kiến trúc ResNet18 (weights=None)**
- Tầng conv đầu tiên + 4 block residual.
- Loại bỏ fc gốc, thêm `nn.Linear(in_fea,1)`.
- Kết quả logits $\rightarrow$ sigmoid để ra xác suất.

c) **Hàm mất mát và optimizer**
- `BCEWithLogitsLoss` kết hợp Sigmoid + BCE.
- Adam $lr = 1 \times 10^{-4}$, không weight decay.

d) **Quy trình huấn luyện (10 epoch)**
- Batch size=4, loop epoch:
  - Forward, compute loss, backward, `optimizer.step()`.
  - In loss epoch.

e) **Đánh giá và dự đoán**
- Chuyển model.eval(), compute sigmoid.
- Ghi xác suất cho tất cả samples vào CSV.

### B. EfficientNet

*1) Cách tiếp cận:*
- **Transfer Learning:** Khởi tạo mô hình từ weights pretrained trên ImageNet để tận dụng kiến thức đã học trên tập lớn.
- **Domain Shift:** Chuyển Mel-spectrogram $(256 \times 256)$ từ grayscale sang 3 kênh (RGB) để phù hợp với input của EfficientNet.
- **Fine-tuning:** Giữ nguyên các lớp đầu, chỉ thay classifier cuối và chỉnh một số block sâu nhất.
- **Multi-label Classification:** Dùng sigmoid cho từng đầu ra và `BCEWithLogitsLoss` để hỗ trợ nhiều loài xuất hiện cùng lúc.

*2) Kiến trúc EfficientNet:*
- **Backbone:** EfficientNet-B0 (hoặc B1–B4 tuỳ cấu hình) với compound scaling.
- **Components:**
  - *Stem:* Conv2d + BatchNorm + Swish activation.
  - *MBConv Blocks:* Depthwise conv + SE module + Skip connection.
  - *Head:* Conv2d + BatchNorm + Swish + AdaptiveAvgPool2d.
  - *Classifier:* Dropout($p = 0.2$) + Linear layer $\rightarrow C$ outputs.
- **Activation:** Sigmoid trên từng đầu ra để tạo xác suất.

*3) Cải tiến so với Baseline CNN:*
- Áp dụng pretrained weights giúp hội tụ nhanh và né overfitting.
- MBConv + SE giúp focus kênh quan trọng.
- AdaptiveAvgPool2d ổn định đầu ra và giảm số tham số.
- Hỗ trợ phân loại multi-label .

*4) Huấn luyện và Đánh giá:*
- **Chia dữ liệu:** 80% train, 20% validation, sử dụng 5-fold cross-validation.
- **Batch size:** 32; **Epochs mỗi fold:** 10;
- **Fold:** 5; **Optimizer:** AdamW.
- **Scheduler:** Cosine Annealing.
- **Augmentation:** SpecAugment, Mixup, Random Cropping.
- **Metrics đánh giá:**
  - Macro mAP (mean Average Precision)
  - Macro F1-score
  - Label-weighted ROC AUC

*5) Kết quả:*
- AUC-ROC (validation) đạt khoảng 0.85–0.90, so với baseline 0.60.
- Soft voting giữa các fold giúp cải thiện độ ổn định của dự đoán.

### C. Cải tiến EfficientNet

Ban đầu, mô hình sử dụng hàm mất mát là `Logit sLoss()` cho tất cả đầu ra. Tuy nhiên, do phân bố nhãn không đều và yêu cầu ưu tiên khó nên cải tiến bằng cách sử dụng hàm mất mát mới là Focal Loss:

- **Logits Loss:**

$$L_{\text{LL}} = -\big[y \log(\hat{y}) + (1-y) \log(1-\hat{y})\big]. \quad (1)$$

- **Focal Loss:**

$$L_{\text{FL}} = -\alpha\,(1-p_t)^{\gamma} \log(p_t), \quad (2)$$

$$p_t = \begin{cases} \hat{y}, & y = 1, \\ 1-\hat{y}, & y = 0, \end{cases} \quad (3)$$

với $\alpha = 0.25$ và $\gamma = 2.0$.
- **Kết hợp hai thành phần:**

$$L = \lambda_{\text{BCE}}\, L_{\text{BCE}} + \lambda_{\text{FL}}\, L_{\text{FL}}, \quad (4)$$

$$\lambda_{\text{BCE}} = 0.6, \quad \lambda_{\text{FL}} = 1.4. \quad (5)$$

- **Triển khai:** Lớp `FocalLossBCE(torch.nn.Module)` định nghĩa:
  - Các tham số: `alpha`, `gamma`, `bce_weight`, `focal_weight`.
  - Sử dụng `BCEWithLogitsLoss` và `sigmoid_focal_loss` (`torchvision.ops`).
  - Trả về: `bce_weight` $\times L_{\text{BCE}} +$ `focal_weight` $\times L_{\text{FL}}$.
- **Mục tiêu:**
  - Giảm thiểu mất cân bằng giữa các lớp.
  - Ưu tiên học các mẫu có dự đoán kém hơn (hard examples).
- **Ý nghĩa:**
  - Focal Loss điều chỉnh trọng số gradient cho các mẫu khó.

  - BCE giữ tính ổn định khi trả về gradient cho các mẫu dễ.
  - Kết hợp giúp mô hình cân bằng giữa học sâu mẫu khó và ổn định tổng thể.

## IV. Kết quả

*1) Kết quả sau khi Submit:*

- **CNN:** 0.520 (submission.csv).
- **EfficientNet:** 0.787 .
- **EfficientNet**$_{Focal_loss}$ : 0.82

## V. Tổng kết

chúng tôi rút ra các nhận xét như sau:

- **Nhận diện đặc trưng mạnh**
  - Các loài có đặc trưng âm thanh cố định và rõ ràng (e.g., *Great Tinamou*, *Clay-colored Thrush*) được mô hình CNN cơ bản và EfficientNet dễ dàng phân biệt với độ chính xác cao (>95%).
- **Những thách thức với âm thanh chồng lấp và nhiễu**
  - Các loài có tín hiệu giống nhau hoặc xuất hiện cùng lúc (e.g., *Wrens*, *Tanagers*) gây nhầm lẫn, yêu cầu mạng học sâu mạnh mẽ để tách đặc trưng. - Nhiễu từ môi trường có thể làm nhiễu; augmentation (SpecAugment, noise injection) đã cải thiện 3% false positive nhưng vẫn còn nhiễu.
- **Ảnh hưởng của label imbalance**
  - Một số loài hiếm xuất hiện ít trong tập huấn luyện khiến mô hình khó học, dễ bị lệch bias về các loài phổ biến. - Sử dụng loss hàm kết hợp FocalLossBCE giúp cải thiện 2–4% F1-score cho các lớp thiểu số .
- **Giới hạn về dữ liệu**
  - Dataset chính chưa đủ đa dạng về vùng địa lý và mùa vụ; việc mở rộng thêm soundscape từ nhiều khu vực sẽ giúp mô hình khái quát tốt hơn.
- **Hiệu năng tính toán và triển khai**
  - EfficientNet-B0 mất 7 phút/epoch trên GPU, phù hợp thử nghiệm nhưng khi triển khai trên edge device có thể cân nhắc phiên bản nhỏ hơn (e.g., B0-lite) hoặc pruning, quantization.
- **Hướng cải tiến tương lai**

  a) Kết hợp mô hình attention (Transformer) để nhắm vào vùng quan trọng.
  b) Triển khai multi-instance learning để tận dụng cả spectrogram dài (soundscape) thay vì chỉ ảnh cắt ngắn.
  c) Khai thác dữ liệu không nhãn (unlabeled) qua self-supervised learning để học đặc trưng chung.
  d) Tích hợp metadata (toa độ, thời gian trong ngày) cùng thông tin môi trường để cải thiện dự đoán phân phối loài.

Đây là những kết quả và báo cáo chính, đưa ra cách phát triển nhằm nâng cao hiệu quả mô hình trong các ứng dụng thực tế của BirdCLEF 2025.

*A. Contributions*

- **Vũ Văn Phong (30%)**
  - Chuẩn bị và tiền xử lý dữ liệu: đọc metadata, padding/cropping, sinh Mel-spectrogram (15%).
  - Thiết kế, triển khai và thử nghiệm mô hình Baseline CNN: kiến trúc, training loop, evaluation (10%).
  - Soạn thảo phần mô tả kỹ thuật và code snippets trong Overleaf cho các mục tiền xử lý và baseline (5%).
- **Lê Anh Tiến (70%)**
  - Phân tích nâng cao và cải tiến mô hình: xây dựng EfficientNet, transfer learning, fine-tuning, loss function (25%).
  - Huấn luyện, đánh giá và tối ưu: data augmentation, scheduler, stratified K-fold, thu thập metrics (25%).
  - Soạn thảo phần phân tích model EfficientNet, thảo luận kết quả và hướng phát triển trong Overleaf (10%).
  - Tích hợp, kiểm thử toàn bộ tài liệu trên Overleaf và hoàn thiện phần khác (10%).

## Tài liệu


[1] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in Proc. ICLR, 2019.
[2] S. Hershey *et al.*, "CNN Architectures for Large-Scale Audio Classification," in Proc. ICASSP, 2017.
[3] BirdCLEF 2025, "BirdCLEF 2025 Competition," Kaggle, 2025. [Online]. Available: https://www.kaggle.com/competitions/birdclef-2025/.