ĐẠI HỌC QUỐC GIA TPHCM

TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

KHOA CÔNG NGHỆ THÔNG TIN

---

# Báo cáo tổng kết

**Đề tài: Nghiên cứu, phân tích và chạy thử nghiệm mô hình AST trong bài báo "AST: Audio Spectrogram Transformer"**

---

**Môn học: Nhập môn học máy**

*Sinh viên thực hiện:*

Tô Hữu Danh (23127336)

Nguyễn Đăng Hưng (23127050)

Lê Phú Cường (23127164)

Nguyễn Bá Đăng Khoa (23127392)

*Giáo viên hướng dẫn:*

Thầy Võ Nhật Tân

Ngày 19 tháng 12 năm 2025

# Mục lục

# Danh sách bảng

# Danh sách hình vẽ

# 1 Giới thiệu

Lĩnh vực phân loại âm thanh (Audio Classification) đóng vai trò then chốt trong sự phát triển của các hệ thống trí tuệ nhân tạo hiện đại. Trong nhiều thập kỷ, các phương pháp tiếp cận từ mô hình thống kê GMM-HMM đến mạng nơ-ron tích chập (CNN) và các biến thể lai ghép (CNN-Attention Hybrid [1]) đã đặt nền móng vững chắc cho việc trích xuất đặc trưng từ tín hiệu âm thanh. Tuy nhiên, các kiến trúc dựa trên tích chập vẫn đối mặt với thách thức cố hữu trong việc nắm bắt các mối quan hệ ngữ cảnh toàn cục do giới hạn về vùng tiếp nhận cục bộ (local receptive field), hạn chế khả năng biểu diễn các chuỗi tín hiệu phức tạp.

Sự xuất hiện của kiến trúc Transformer, đặc biệt là thành công của mô hình Vision Transformer (ViT) [2] trong thị giác máy tính, đã mở ra một hướng đi đột phá: xử lý dữ liệu không gian dưới dạng chuỗi các mảnh ghép (patches). Kế thừa tư duy này, mô hình **Audio Spectrogram Transformer (AST)** [3] đã được đề xuất như một giải pháp tiên phong loại bỏ hoàn toàn sự phụ thuộc vào tích chập (convolution-free). Bằng cơ chế Tự chú ý (Self-Attention), AST không chỉ giải quyết bài toán mô hình hóa sự phụ thuộc thời gian - tần số ở phạm vi toàn cục mà còn tận dụng hiệu quả tri thức chuyển giao (Transfer Learning) từ các bộ dữ liệu quy mô lớn như ImageNet và AudioSet để đạt hiệu suất vượt trội.

Trong phạm vi báo cáo này, chúng tôi tập trung nghiên cứu cơ chế hoạt động của kiến trúc AST và thực hiện quy trình thực nghiệm tinh chỉnh (fine-tuning) mô hình AST – vốn được huấn luyện trước trên AudioSet – để giải quyết bài toán phân loại âm thanh môi trường trên tập dữ liệu UrbanSound8K. Nghiên cứu nhằm mục đích kiểm chứng khả năng tổng quát hóa và hiệu quả của phương pháp học chuyển giao khi áp dụng các mô hình Transformer tiên tiến vào các bài toán cụ thể với tài nguyên dữ liệu giới hạn.

# 2 Các công trình liên quan

Lịch sử phát triển của bài toán phân loại âm thanh và nhận dạng sự kiện âm thanh (Audio Event Detection - AED) là một quá trình tiến hóa liên tục nhằm tìm kiếm các biểu diễn đặc trưng (feature representations) hiệu quả hơn. Quá trình này có thể được chia thành ba giai đoạn chính: từ các mô hình thống kê dựa trên đặc trưng thủ công, đến các mạng nơ-ron tích chập (CNN) và các biến thể lai ghép, và gần đây nhất là sự trỗi dậy của kiến trúc Transformer thuần túy.

## 2.1 Phương pháp thống kê và mô hình GMM-HMM

Trong giai đoạn đầu, các hệ thống nhận dạng âm thanh chủ yếu dựa vào các đặc trưng được thiết kế thủ công (hand-crafted features) như Mel-Frequency Cepstral Coefficients (MFCCs) hay Filterbanks. Mô hình tiêu biểu nhất trong giai đoạn này là sự kết hợp giữa **Mô hình Hỗn hợp Gaussian (Gaussian Mixture Models - GMM)** và **Mô hình Markov Ẩn (Hidden Markov Models - HMM)** [4].

Về mặt lý thuyết, GMM được sử dụng để mô hình hóa phân phối xác suất của các đặc trưng quang phổ tại mỗi khung thời gian (acoustic modeling), trong khi HMM chịu trách nhiệm mô hình hóa sự phụ thuộc và chuyển đổi trạng thái theo trình tự thời gian (temporal modeling). Mặc dù GMM-HMM đã đặt nền móng vững chắc cho ngành xử lý tiếng nói, chúng tồn tại những hạn chế cố hữu:

- **Phụ thuộc vào trích xuất đặc trưng:** Hiệu suất mô hình bị giới hạn bởi chất lượng của các đặc trưng thủ công, vốn không thể nắm bắt hết sự phức tạp của tín hiệu âm thanh thực tế.
- **Giả định độc lập:** HMM dựa trên giả định Markov, cho rằng trạng thái hiện tại chỉ phụ thuộc vào trạng thái liền trước, do đó gặp khó khăn trong việc nắm bắt các phụ thuộc dài hạn (long-term dependencies) trong chuỗi tín hiệu.

## 2.2 Mạng Nơ-ron Tích chập (CNN) và Kiến trúc Lai ghép (Hybrid Models)

Sự ra đời của AlexNet [5] vào năm 2012 đã đánh dấu sự chuyển dịch sang kỷ nguyên học sâu (Deep Learning). Các nhà nghiên cứu bắt đầu tiếp cận âm thanh dưới dạng hình ảnh thông qua biểu đồ

phổ (Spectrogram), cho phép áp dụng các kiến trúc Mạng nơ-ron tích chập (CNN) mạnh mẽ như VGG hay ResNet để tự động học các đặc trưng từ dữ liệu thay vì thiết kế thủ công.
CNN sở hữu thiên kiến quy nạp (inductive bias) mạnh mẽ, bao gồm tính cục bộ (locality) và tính bất biến tịnh tiến (translation equivariance), giúp chúng rất hiệu quả trong việc phát hiện các mẫu hình tần số cục bộ. Tuy nhiên, CNN thuần túy gặp hạn chế trong việc mô hình hóa chuỗi thời gian toàn cục do giới hạn của vùng tiếp nhận (receptive field). Để khắc phục, các kiến trúc Lai ghép (Hybrid Models) đã được đề xuất:

- **CRNN (Convolutional Recurrent Neural Networks):** Kết hợp CNN (để trích xuất đặc trưng không gian) với RNN hoặc LSTM (để mô hình hóa chuỗi thời gian).
- **CNN-Attention:** Thay thế hoặc bổ sung cơ chế RNN bằng cơ chế Chú ý (Attention Mechanism) để cho phép mô hình tập trung vào các đoạn âm thanh quan trọng và tổng hợp thông tin tốt hơn.

Mặc dù các mô hình lai ghép này đã đạt được những kết quả vượt trội, chúng vẫn chịu ràng buộc bởi cấu trúc tích chập: khả năng nắm bắt thông tin toàn cục chỉ đạt được ở các lớp rất sâu hoặc thông qua các phép toán gộp (pooling) làm mất mát thông tin chi tiết.

## 2.3 Kiến trúc Transformer và Vision Transformer (ViT)

Kiến trúc Transformer, ban đầu được giới thiệu cho các tác vụ xử lý ngôn ngữ tự nhiên, dựa hoàn toàn vào cơ chế Tự chú ý (Self-Attention) để mô hình hóa các mối quan hệ toàn cục trong chuỗi dữ liệu. Dosovitskiy et al. (2020) đã đề xuất Vision Transformer (ViT) [2], minh chứng rằng một kiến trúc Transformer thuần túy có thể đạt hiệu suất vượt trội trong thị giác máy tính bằng cách chia hình ảnh thành chuỗi các mảnh ghép (patches) cố định (ví dụ: $16 \times 16$) và xử lý chúng tương tự như các từ trong câu.
Khác biệt cốt lõi của ViT so với CNN nằm ở Vùng tiếp nhận toàn cục (Global Receptive Field). Ngay từ lớp đầu tiên, cơ chế Self-Attention cho phép mỗi mảnh ghép tham chiếu thông tin từ tất cả các mảnh ghép khác, giúp mô hình nắm bắt cấu trúc tổng thể của dữ liệu ngay lập tức mà không cần trải qua nhiều lớp tích chập phân cấp. Hơn nữa, ViT có thiên kiến quy nạp yếu hơn, cho phép mô hình linh hoạt hơn trong việc học các mối quan hệ phức tạp từ các tập dữ liệu quy mô lớn.

## 2.4 Audio Spectrogram Transformer (AST)

Kế thừa thành tựu của ViT, Gong et al. (2021) đã đề xuất mô hình Audio Spectrogram Transformer (AST) [3]. Đây là kiến trúc đầu tiên áp dụng cơ chế Transformer không tích chập (convolution-free) cho bài toán phân loại âm thanh.

AST xử lý biểu đồ phổ âm thanh đầu vào (kích thước $128 \times 100t$) bằng cách chia nhỏ thành chuỗi các mảnh $16 \times 16$ có sự chồng lấn (overlap). Các mảnh này được chiếu tuyến tính thành các vector nhúng (embeddings), cộng với thông tin vị trí (positional embedding) và đưa qua bộ mã hóa Transformer chuẩn. Điểm đột phá của AST nằm ở khả năng tận dụng tri thức chuyển giao (Transfer Learning) từ miền hình ảnh (ImageNet) sang miền âm thanh. Các tác giả đã đề xuất các kỹ thuật thích ứng hiệu quả như trung bình hóa trọng số kênh và nội suy vị trí để giải quyết sự khác biệt về chiều dữ liệu giữa hai miền.

Các thực nghiệm trên AudioSet và ESC-50 cho thấy AST không chỉ vượt qua các mô hình CNN-Attention lai ghép tốt nhất (SOTA) mà còn chứng minh khả năng hội tụ nhanh và hiệu quả dữ liệu vượt trội nhờ vào cơ chế chú ý toàn cục và chiến lược khởi tạo trọng số thông minh.

# 3 Cài đặt mô hình

Mô hình Audio Spectrogram Transformer (AST) được xây dựng dựa trên nền tảng kiến trúc DeiT (Data-efficient Image Transformers) để tận dụng các trọng số pre-trained hiệu quả từ ImageNet. Chi tiết cài đặt như sau: [3]

**Đầu vào và tiền xử lý:**

- **Đặc trưng:** Tín hiệu âm thanh (waveform) được chuyển đổi thành log-Mel spectrogram 128 chiều.
- **Tham số STFT:** Cửa sổ Hamming 25 ms, bước trượt 10 ms; chuẩn hoá đặc trưng trước khi đưa vào mạng.
- **Kích thước:** Với đoạn âm thanh dài $t$ giây, số khung thời gian xấp xỉ $100t$, tạo ra spectrogram kích thước $128 \times 100t$.

**Phân mảnh (Patching):**

- Spectrogram được chia thành các patch kích thước $16 \times 16$.
- Áp dụng cơ chế chồng lấn (overlap) với stride = 10 trên cả hai trục thời gian và tần số (tương đương vùng chồng lấn 6 đơn vị).
- Số patch theo trục tần số cố định là 12; theo trục thời gian là $\left\lfloor \frac{100t-16}{10} \right\rfloor + 1$.
- Tổng số patch:

$$N \;=\; 12 \times \left( \left\lfloor \frac{100t - 16}{10} \right\rfloor + 1 \right).$$

**Kiến trúc Transformer và Token đặc biệt:** Mô hình sử dụng backbone là ViT (biến thể DeiT-Base) với cấu hình: embedding $d = 768$, 12 lớp encoder, 12 đầu attention.

- **Token đặc biệt:** Do sử dụng kiến trúc DeiT, chuỗi đầu vào được bổ sung hai token đặc biệt ở đầu: Token phân loại (`[CLS]`) và Token chưng cất (`[DIST]`).
- **Chuỗi đưa vào Encoder:**

$$Input = [\texttt{CLS}, \texttt{DIST}, Patch_1, Patch_2, \ldots, Patch_N] + PosEmbed$$

- **Mã hóa vị trí:** Sử dụng positional embedding có thể học được. Trong code gốc, vector này được nội suy (interpolate) động để phù hợp với chiều dài thay đổi của âm thanh đầu vào.

**Đầu ra và huấn luyện:**

- Lấy vector của [CLS] làm biểu diễn và đưa qua đầu phân loại tuyến tính.
- Bài toán gốc (AudioSet): đa nhãn, sử dụng sigmoid ở đầu ra và Binary Cross-Entropy (BCE) làm hàm mất mát.
- Pretrain: Khởi tạo từ ViT đã được pretrain trên ImageNet hoặc Imagenet + AudioSet; sau đó fine-tune trên tập target.

**Đầu ra và huấn luyện:**

- **Biểu diễn đầu ra (Pooling):** Đặc trưng cuối cùng của đoạn âm thanh không chỉ dùng token `[CLS]` mà là trung bình cộng của token `[CLS]` và `[DIST]`:

$$h_{final} = \frac{h_{[CLS]} + h_{[DIST]}}{2}$$

- **Lớp phân loại:** Vector $h_{final}$ được chuẩn hóa (LayerNorm) trước khi đưa qua lớp tuyến tính (Linear Head) để dự đoán nhãn.
- **Hàm mất mát:** Sử dụng Binary Cross-Entropy (BCE) cho bài toán đa nhãn (AudioSet).

# 4 Mô tả tập dữ liệu

Trong nghiên cứu này, nhóm sử dụng tập dữ liệu UrbanSound8K, một bộ dữ liệu âm thanh môi trường đô thị được xây dựng để phục vụ các bài toán phân loại các đoạn âm thanh ngắn trong đời sống hằng ngày. UrbanSound8K bao gồm 8.732 đoạn âm thanh (audio clips) đã được gán nhãn, với thời lượng không quá 4 giây cho mỗi đoạn. Các đoạn âm thanh này được trích xuất từ các bản ghi dài hơn và được chú thích cẩn thận.

Toàn bộ dữ liệu được chia thành 10 fold (fold1 đến fold10), tuân theo chiến lược 10-fold cross-validation được đề xuất sẵn bởi tác giả bộ dữ liệu. Cách chia này được thiết kế nhằm đảm bảo người dùng có thể đánh giá mô hình một cách công bằng và nhất quán, đồng thời hạn chế hiện tượng data leakage (rò rỉ dữ liệu giữa tập huấn luyện và tập kiểm thử). Trong mỗi fold, các đoạn âm thanh thuộc nhiều lớp khác nhau được phân bố một cách tương đối cân bằng.

UrbanSound8K được gán nhãn theo 10 loại âm thanh (10 classes) thường gặp trong môi trường đô thị, ví dụ như tiếng ô tô (car horn), tiếng chó sủa (dog bark), tiếng khoan (drilling), tiếng còi cứu hỏa (siren), tiếng máy nổ (engine idling), v.v. Mỗi bản ghi được mô tả bởi các thông tin meta kèm theo như: tên file, fold, mã lớp (class ID) và tên lớp (class name). Các thông tin này được lưu trong file metadata UrbanSound8K.csv, giúp việc truy vấn, lọc và xây dựng tập train/test trở nên thuận tiện.

Về mặt tổ chức thư mục, UrbanSound8K được cấu trúc như sau:

- Thư mục audio/ chứa 10 thư mục con: fold1/, fold2/, . . . , fold10/.
- Mỗi thư mục foldX/ chứa các file âm thanh định dạng .wav tương ứng với fold đó.
- Thư mục metadata/ chứa file UrbanSound8K.csv, trong đó mỗi dòng tương ứng với một đoạn âm thanh kèm nhãn và thông tin mô tả.

Tập dữ liệu UrbanSound8K có dung lượng khoảng 6 GB, bao gồm cả các file âm thanh và metadata. Với quy mô vừa phải nhưng đa dạng về loại âm thanh và điều kiện thu, UrbanSound8K là lựa chọn phù hợp cho các thí nghiệm *fine-tuning* mô hình Audio Spectrogram Transformer (AST) trong bối cảnh phân loại âm thanh môi trường. Đồng thời, đây cũng là một bộ dữ liệu phổ biến trong cộng đồng, cho phép so sánh kết quả với nhiều mô hình khác (CNN, CRNN, hay các mô hình transformer khác) đã được công bố trước đó.

# 5 Cài đăt thử nghiệm của tác giả [6]

Để đánh giá hiệu quả của mô hình Audio Spectrogram Transformer (AST), tác giả thiết lập quy trình huấn luyện và kiểm thử với các cấu hình chi tiết như sau.

## 5.1 Môi trường tối ưu hoá và huấn luyện

Mô hình được huấn luyện theo phương pháp học giám sát (supervised learning) sử dụng thuật toán tối ưu hóa Adam. Các tham số của bộ tối ưu hóa được thiết lập như sau:

- **Trọng số suy giảm (Weight decay):** $5 \times 10^{-7}$.
- **Các hệ số Beta:** $\beta_1 = 0.95$, $\beta_2 = 0.999$.

Chiến lược khởi tạo trọng số (Initialization) tận dụng mô hình đã được pre-trained trên ImageNet (đối với tất cả các tập dữ liệu) để tăng cường khả năng trích xuất đặc trưng và hội tụ nhanh hơn.

## 5.2 Xử lý dữ liệu và tăng cường (Augmentation)

Dữ liệu đầu vào là Log-Mel Spectrogram được chuẩn hóa (Normalization) bằng cách trừ đi giá trị trung bình (Mean) và chia cho độ lệch chuẩn (Std) của toàn bộ tập dữ liệu.
Để giảm thiểu hiện tượng quá khớp (overfitting), tác giả áp dụng các kỹ thuật tăng cường dữ liệu mạnh mẽ ngay trong quá trình tải dữ liệu (online augmentation):

- **SpecAugment:** Áp dụng mặt nạ tần số (Frequency Masking) và mặt nạ thời gian (Time Masking) với kích thước tối đa tùy thuộc vào từng tập dữ liệu.
- **Mixup:** Trộn lẫn hai mẫu dữ liệu ngẫu nhiên với hệ số $\lambda$ được lấy mẫu từ phân phối Beta hoặc Uniform, giúp mô hình học được các đặc trưng lai giữa các lớp.
- **Noise Augmentation:** Thêm nhiễu ngẫu nhiên vào spectrogram (chỉ áp dụng cho tập Speech-Commands).

## 5.3 Cấu hình chi tiết cho từng bộ dữ liệu

Các siêu tham số (Hyperparameters) cụ thể cho từng bài toán thực nghiệm được thiết lập dựa trên các kịch bản chuẩn (`run.sh`, `run_sc.sh`, `run_esc.sh`) như bảng dưới đây:

| **Tham số** | **AudioSet (Full)** | **SpeechCommands (v2)** | **ESC-50** |
|---|---|---|---|
| **Kích thước đầu vào** | 1024 frames (∼10s) | 128 frames (∼1s) | 512 frames (∼5s) |
| **Batch Size** | 12 | 128 | 48 |
| **Learning Rate (LR)** | $1 \times 10^{-5}$ | $2.5 \times 10^{-4}$ | $1 \times 10^{-5}$ |
| **Số Epochs** | 5 | 30 | 25 |
| **LR Scheduler** | MultiStepLR (Giảm 0.5 mỗi epoch từ epoch 2) | MultiStepLR (Giảm 0.85 mỗi epoch từ epoch 5) | MultiStepLR (Giảm 0.85 mỗi epoch từ epoch 5) |
| **Warmup** | Có | Không | Không |
| **SpecAugment** | Freq Mask: 48, Time Mask: 192 | Freq Mask: 48, Time Mask: 48 | Freq Mask: 24, Time Mask: 96 |
| **Mixup ($\alpha$)** | 0.5 | 0.6 | 0 |
| **Hàm mất mát** | BCEWithLogitsLoss | BCEWithLogitsLoss | CrossEntropyLoss |
| **Độ đo chính** | mAP | Accuracy | Accuracy |

Bảng 1: Bảng tham số thực nghiệm cho các bộ dữ liệu AudioSet, SpeechCommands và ESC-50

**Lưu ý đặc biệt cho AudioSet:** Đối với tập dữ liệu AudioSet (cấu hình Full), tác giả sử dụng kỹ thuật **Weight Averaging** (trung bình trọng số mô hình) từ epoch 1 đến epoch 5 để tạo ra mô hình cuối cùng ổn định hơn.

**Thông số chuẩn hoá:** Các giá trị trung bình, độ lệch chuẩn được sử dụng cho chuẩn hoá đầu vào:

- AudioSet: Mean - 4.268, Std 4.569.
- SpeechCommands: Mean - 6.846, Std 5.565.
- ESC-50: Mean - 6.627, Std 5.358.

# 6 Thử nghiệm của nhóm

## 6.1 Môi trường thực nghiệm

Các thí nghiệm được triển khai trong môi trường notebook trực tuyến có hỗ trợ GPU với nền tảng CUDA (Kaggle Notebook). Mô hình được hiện thực bằng thư viện PyTorch (phiên bản 2.x), kết hợp cùng các thư viện torchaudio, timm, librosa và scikit-learn để xử lý tín hiệu và xây dựng pipeline huấn luyện. Các phép tính trên GPU được tăng tốc bằng cơ chế mixed precision training thông qua mô-đun torch.amp.autocast và GradScaler, giúp giảm dung lượng bộ nhớ và rút ngắn thời gian huấn luyện.
Dữ liệu UrbanSound8K được mount sẵn tại thư mục /kaggle/input và toàn bộ mã nguồn, checkpoint của mô hình được lưu tại /kaggle/working.

## 6.2 Tiền xử lý tín hiệu âm thanh

Mọi đoạn âm thanh trong UrbanSound8K đều được chuyển về dạng log Mel filterbank (fbank) trước khi đưa vào mô hình Audio Spectrogram Transformer (AST). Quy trình tiền xử lý gồm các bước:

- Tải tín hiệu âm thanh bằng torchaudio với dạng tensor (channels, time).
- Nếu tín hiệu có nhiều kênh (stereo), chuyển về mono bằng cách lấy trung bình trên trục kênh.
- Chuẩn hóa tần số lấy mẫu về 16 kHz bằng phép nội suy lại (resampling).
- Cắt hoặc pad tín hiệu về đúng 10 giây, tương ứng với một số mẫu cố định.
- Trích xuất đặc trưng log Mel filterbank với 128 băng tần Mel, window 25 ms, bước trượt 10 ms, thu được khoảng 1024 khung thời gian.
- Nếu số khung thời gian nhỏ hơn 1024 thì pad thêm khung 0; nếu lớn hơn thì cắt bớt cho vừa 1024 khung.
- Chuẩn hóa đặc trưng theo công thức $(x - \mu)/(2\sigma)$ với $\mu = -4.2677$ và $\sigma = 4.5690$, nhằm đưa về miền giá trị phù hợp với mô hình AST gốc.

Kết quả mỗi đoạn âm thanh là một tensor kích thước (T, F) = (1024, 128), sau đó được đưa vào mô hình dưới dạng batch.

## 6.3 Cấu hình mô hình Audio Spectrogram Transformer

Trong tất cả các thí nghiệm, nhóm sử dụng cùng một kiến trúc Audio Spectrogram Transformer (AST) với cấu hình base384, stride thời gian và tần số đều bằng 10. Với đầu vào 128 băng tần Mel và 1024 khung thời gian, mô hình tạo ra 1212 patch spectrogram. Phần đầu ra của backbone là một vector đặc trưng có chiều 768, sau đó được đưa qua một lớp phân loại tuyến tính (mlp head) ánh xạ về 10 lớp tương ứng với 10 loại âm thanh của UrbanSound8K.
Trên cơ sở kiến trúc chung này, nhóm xây dựng hai cấu hình tiền huấn luyện khác nhau:

### 6.3.1 Cấu hình AST-S (AST-ImageNet)

Trong kịch bản thứ nhất, mô hình AST được khởi tạo từ các trọng số đã được huấn luyện trước trên ImageNet thông qua tham số imagenet_pretrain = True và audioset_pretrain = False trong hàm khởi tạo ASTModel. Toàn bộ backbone vì vậy kế thừa khả năng trích xuất đặc trưng từ ảnh (được ánh xạ sang miền spectrogram), nhưng không sử dụng thêm bất kì checkpoint nào từ AudioSet.
Lớp phân loại gốc của AST (với 527 nút đầu ra) được thay thế bằng một lớp tuyến tính mới có 10 nút, khởi tạo ngẫu nhiên, để phù hợp với bài toán phân loại 10 lớp trên UrbanSound8K. Cấu hình này được ký hiệu là AST-S.

### 6.3.2 Cấu hình AST-P (AST-ImageNet, AudioSet)

Trong kịch bản thứ hai, mô hình AST được khởi tạo sao cho tương thích với checkpoint đã fine-tune trên AudioSet (audioset_0.4593.pth). Cụ thể, mô hình được tạo với label_dim = 527, input_tdim = 1024, input_fdim = 128, và cả hai tham số imagenet_pretrain, audioset_pretrain đều đặt False để nhóm chủ động nạp trọng số từ checkpoint AudioSet bên ngoài.
Sau khi tải file trọng số audioset_0.4593.pth, toàn bộ state_dict được nạp vào mô hình, giúp backbone kế thừa trực tiếp khả năng phân biệt âm thanh từ tập AudioSet. Tương tự như cấu hình trước, lớp phân loại 527 chiều được thay bằng một lớp tuyến tính 10 chiều, khởi tạo ngẫu nhiên cho UrbanSound8K. Cấu hình này được ký hiệu là AST-P.
Hai cấu hình trên sử dụng chung toàn bộ pipeline tiền xử lý, kiến trúc backbone, hàm mất mát, optimizer và chiến lược huấn luyện; điểm khác biệt duy nhất nằm ở nguồn trọng số tiền huấn luyện được sử dụng cho backbone.

## 6.4 Chiến lược huấn luyện và đánh giá

Để đánh giá mô hình một cách công bằng, nhóm áp dụng chiến lược 10-fold cross-validation theo đúng cách chia fold có sẵn trong UrbanSound8K:

- Ở mỗi vòng lặp, chọn một fold (từ 1 đến 10) làm tập kiểm thử.
- Chín fold còn lại được gộp lại tạo thành tập huấn luyện.
- Tuyệt đối không trộn mẫu giữa các fold, đảm bảo không xảy ra rò rỉ dữ liệu.

Với mỗi cấu hình mô hình (AST-S và AST-P), nhóm sử dụng cùng một bộ siêu tham số:

- Hàm mất mát: Cross-entropy cho bài toán phân loại đơn nhãn.
- Optimizer: AdamW với learning rate cố định là $1 \times 10^{-5}$.
- Kích thước batch: 8 (được lựa chọn để phù hợp với dung lượng bộ nhớ GPU).
- Số epoch: từ 3 đến 5 epoch cho mỗi fold, tùy theo giới hạn tính toán; trong cả hai cấu hình, mô hình hội tụ khá nhanh nhờ sử dụng trọng số tiền huấn luyện.
- Huấn luyện sử dụng mixed precision (autocast và GradScaler) để giảm chi phí tính toán.

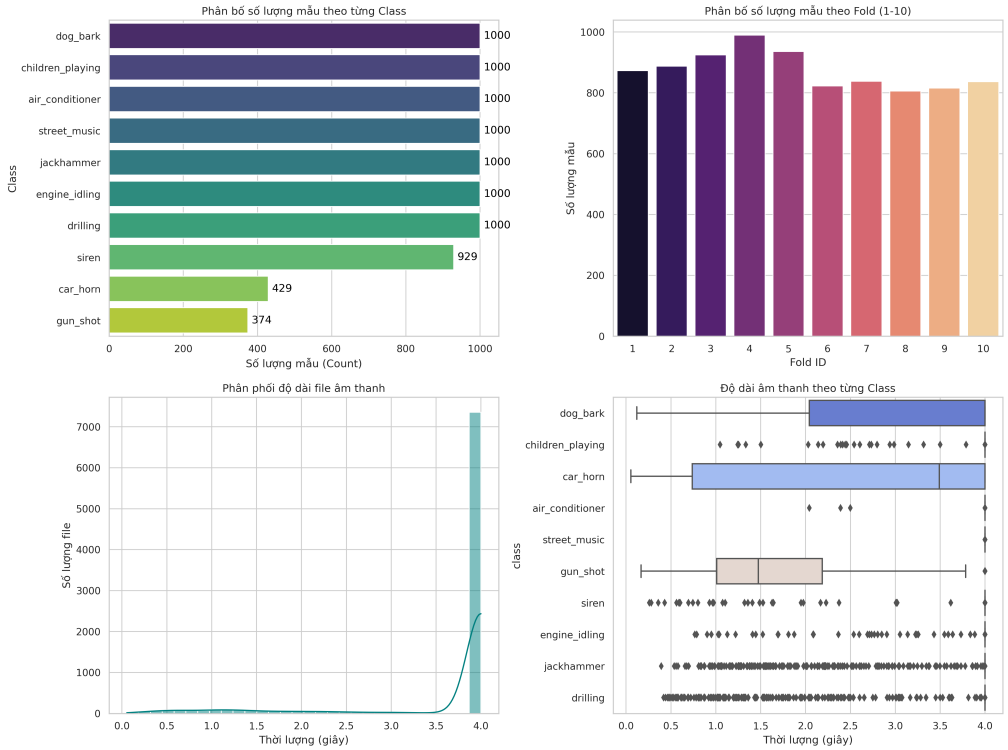
Sau mỗi epoch, mô hình được đánh giá trên tập kiểm thử của fold tương ứng. Nhóm lưu lại checkpoint có độ chính xác cao nhất trên tập kiểm thử (best test accuracy) cho mỗi fold. Các checkpoint này được ghi vào thư mục ast_us8k_checkpoints.
Cuối cùng, để tổng hợp kết quả, nhóm tính toán độ chính xác trung bình và độ lệch chuẩn trên toàn bộ 10 fold, theo công thức:

$$\text{Mean Accuracy} = \frac{1}{10}\sum_{k=1}^{10}\text{Acc}_k, \quad \text{Std} = \sqrt{\frac{1}{10}\sum_{k=1}^{10}(\text{Acc}_k - \text{Mean Accuracy})^2}.$$

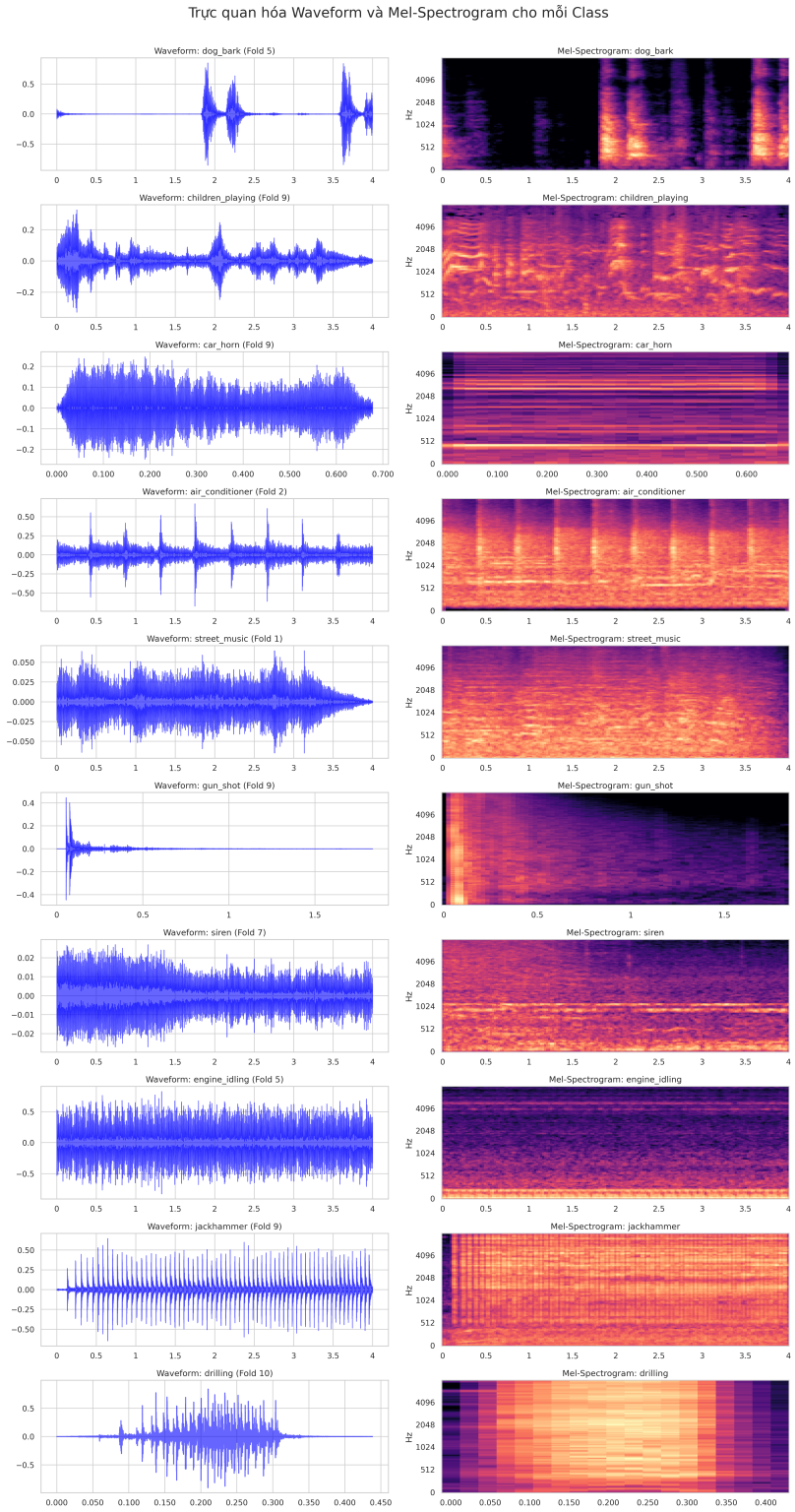
Việc sử dụng cùng một thiết lập huấn luyện và cùng chiến lược cross-validation cho cả AST-S và AST-P giúp việc so sánh trực tiếp tác động của tiền huấn luyện trên AudioSet đối với hiệu năng phân loại âm thanh môi trường trên UrbanSound8K.

## 6.5 Lí do chọn cấu hình

### 6.5.1 Phân tích EDA dữ liệu




Hình 1: Thống kê phân phối mẫu của các lớp




Hình 2: Waveform và spectrogram của các lớp

Kết quả phân tích dữ liệu khám phá (EDA) và rà soát kịch bản huấn luyện (training script) hiện tại cho thấy cấu hình tham số (hyperparameters) đang được thiết lập chưa tương thích tốt với đặc

điểm phân bố của dữ liệu. Điều này dẫn đến sự lãng phí tài nguyên tính toán và nguy cơ mô hình không đạt được điểm hội tụ tối ưu (suboptimal convergence).
Từ các biểu đồ thống kê và trực quan hóa tín hiệu trên, hai đặc điểm cốt lõi của tập dữ liệu UrbanSound8K đã được xác định:

- **Phân bố thời gian (Temporal Distribution):** Biểu đồ histogram cho thấy mật độ tập trung cao của các mẫu dữ liệu ở độ dài 4.0 giây. Rất ít mẫu vượt quá ngưỡng này.
- **Sự mất cân bằng dữ liệu (Class Imbalance):** Tồn tại sự chênh lệch đáng kể về số lượng mẫu giữa các lớp. Cụ thể, các lớp như `car_horn` và `gun_shot` có số lượng mẫu thấp hơn khoảng 60% so với các lớp đa số (như `engine_idling`, `jackhammer`).

Khác với các tập dữ liệu hình ảnh thông thường (nơi các ảnh thường độc lập với nhau), UrbanSound8K có đặc thù về nguồn gốc dữ liệu. Cụ thể, dữ liệu là các file âm thanh ngắn (slices) được cắt ra từ các bản thu âm dài gốc. Nếu chia tập Train/Test một cách ngẫu nhiên (random split), rất có thể các đoạn cắt (slices) từ cùng một bản thu âm gốc sẽ nằm rải rác ở cả tập Train và tập Test, từ đó mô hình sẽ học thuộc lòng đặc trưng nền (background noise) của bản thu gốc thay vì học đặc trưng của loại âm thanh đó. Điều này dẫn đến kết quả kiểm thử cao giả tạo nhưng mô hình thực tế lại hoạt động kém. Việc sử dụng cột fold có sẵn trong file metadata (được thiết kế sao cho các đoạn cắt từ cùng một nguồn luôn nằm chung một fold) là biện pháp kỹ thuật bắt buộc để đảm bảo tính độc lập dữ liệu (Data Independence).
Tuy nhiên, cấu hình huấn luyện hiện tại được đánh giá là chưa tối ưu dựa trên các luận điểm khoa học sau:

**Lãng phí tài nguyên tính toán do padding quá nhiều** Hiện tại, mô hình đang thiết lập độ dài đầu vào là 1024 frames (tương đương khoảng 10 giây). Tuy nhiên, thực tế dữ liệu chỉ dài trung bình 4 giây, điều này buộc hệ thống phải thực hiện kỹ thuật Zero-Padding (chèn các giá trị 0) vào khoảng 60% kích thước của tensor đầu vào. Từ đó, cơ chế Self-Attention của Transformer phải thực hiện tính toán trên một vùng không gian thưa (sparse) chứa ít thông tin hữu ích. Điều này không chỉ gây lãng phí bộ nhớ GPU và thời gian huấn luyện mà còn có thể gây nhiễu cho quá trình trích xuất đặc trưng.

**Sai lệch mô hình do mất cân bằng dữ liệu** Việc không áp dụng các kỹ thuật cân bằng dữ liệu (Data Balancing) trong bối cảnh tập dữ liệu bị lệch (skewed dataset) khiến hàm mất mát (Loss function) sẽ bị chi phối bởi các lớp đa số. Từ đó, mô hình có xu hướng tối ưu hóa độ chính xác toàn cục bằng cách hy sinh khả năng nhận diện các lớp thiểu số (`gun_shot`, `car_horn`), dẫn đến chỉ số F1-Score thấp mặc dù Accuracy có thể cao.

# 7 Đánh giá và phân tích kết quả

## 7.1 Thời gian huấn luyện mô hình (Kaggle Runtime)

Thời gian được ghi nhận trực tiếp từ Runtime hiển thị trên Kaggle Notebook cho từng notebook chạy thực nghiệm. Do Runtime phản ánh wall-clock time của toàn bộ notebook, giá trị này có thể bao gồm cả bước chuẩn bị dữ liệu và đánh giá, và có thể dao động nhẹ do tải hệ thống. Tuy nhiên, các notebook được chạy trên cùng nền tảng Kaggle (GPU) với pipeline nhất quán.
Thí nghiệm gồm 10-fold cross-validation và hai cấu hình: AST-S (pretrain ImageNet) và AST-P (pretrain ImageNet+AudioSet). Do giới hạn thời gian chạy, nhóm chia thành 4 notebook tương ứng với 2 cấu hình $\times$ 2 nhóm fold (1–5 và 6–10).

**Liên kết notebook.**

- AST-S (Folds 1–5) – Notebook
- AST-S (Folds 6–10) – Notebook
- AST-P (Folds 1–5) – Notebook
- AST-P (Folds 6–10) – Notebook

| Cấu hình | Nhóm fold | Runtime (hh:mm:ss) |
|---|---|---|
| AST-S (ImageNet) | 1–5 | 2:35:22 |
| AST-S (ImageNet) | 6–10 | 2:22:01 |
| AST-P (ImageNet+AudioSet) | 1–5 | 2:22:56 |
| AST-P (ImageNet+AudioSet) | 6–10 | 2:29:46 |

Bảng 2: Runtime ghi nhận từ Kaggle Notebook theo cấu hình và nhóm fold.

## 7.2 Kết quả định lượng trên UrbanSound8K

Sau khi huấn luyện và lưu checkpoint cho từng fold, nhóm tiến hành đánh giá lại toàn bộ 10 mô hình trên tập kiểm thử tương ứng của mỗi fold. Quá trình đánh giá được thực hiện riêng cho hai cấu hình:

- AST-P: backbone khởi tạo từ checkpoint đã fine-tune trên AudioSet, sau đó thay đầu ra 10 lớp và fine-tune trên UrbanSound8K.

- AST-S: backbone khởi tạo từ mô hình tiền huấn luyện trên ImageNet, chỉ fine-tune trực tiếp trên UrbanSound8K, không sử dụng checkpoint AudioSet.

Bảng dưới đây tóm tắt kết quả trung bình trên 10 fold:

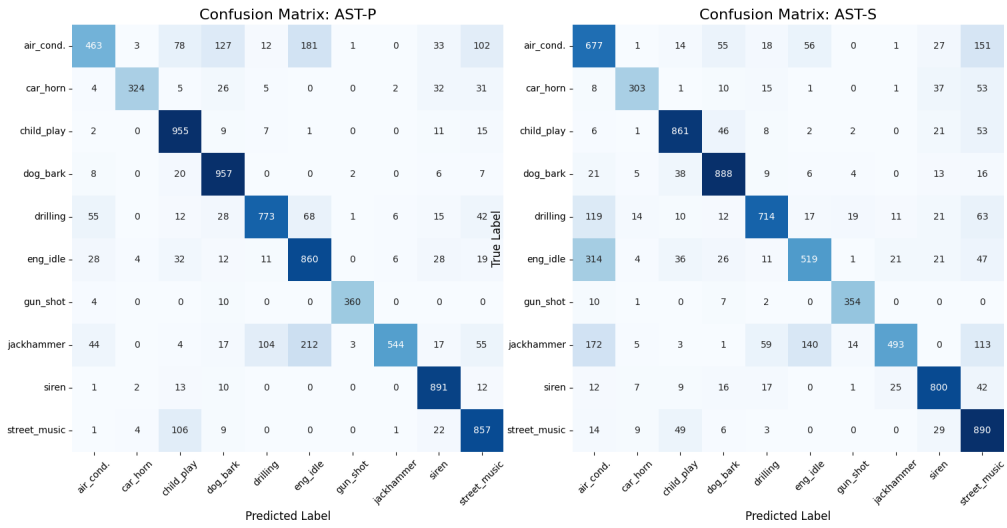
| Cấu hình mô hình | Mean accuracy (%) | Std (%) | Micro accuracy (%) |
|---|---|---|---|
| AST-S | 74.41 | 5.72 | 74.41 |
| AST-P | 80.08 | 6.76 | 80.08 |

Bảng 3: Kết quả 10-fold cross-validation trên UrbanSound8K cho hai cấu hình AST.

Đối với cấu hình AST-P, độ chính xác trên từng fold dao động từ khoảng 65.62% (fold khó nhất) đến 90.32% (fold tốt nhất), với giá trị trung bình 80.08% và độ lệch chuẩn 6.76%. Micro accuracy tính trực tiếp từ confusion matrix tổng hợp trên toàn bộ dữ liệu cũng đạt 80.08%, trùng khớp với giá trị trung bình theo fold, cho thấy quá trình tổng hợp và đánh giá là nhất quán.
Trong khi đó, cấu hình AST-S đạt độ chính xác trung bình 74.41% với độ lệch chuẩn 5.72%. Các fold dao động trong khoảng từ 65.95% đến 83.33%, micro accuracy tính từ confusion matrix tổng là 74.41%. Như vậy, so với AST-S, mô hình AST-P cải thiện khoảng 5.7 điểm phần trăm về độ chính xác trung bình trên UrbanSound8K.

## 7.3 Phân tích confusion matrix

Để quan sát chi tiết hơn cách mô hình nhầm lẫn giữa các lớp, nhóm tổng hợp confusion matrix trên toàn bộ 10 fold cho từng cấu hình. Thứ tự các lớp trong bảng là: air_conditioner, car_horn, children_playing, dog_bark, drilling, engine_idling, gun_shot, jackhammer, siren, street_music.




Hình 3: Ma trận nhầm lẫn thể hiện khả năng dự đoán của 2 cấu hình

Từ hai confusion matrix trong Hình 3, nhóm rút ra một số nhận xét định tính về hành vi của mô hình.

**Các lớp dễ phân biệt.** Các lớp mang tính sự kiện ngắn, có đặc trưng phổ tần rõ ràng như gun_shot, car_horn, dog_bark và children_playing đạt độ chính xác cao ở cả hai cấu hình. Đặc biệt, lớp gun_shot gần như không bị nhầm lẫn sang các lớp khác, phản ánh khả năng học tốt các mẫu âm thanh có biên dạng thời gian và phổ tần đặc trưng.

**Các lớp khó, dễ gây nhầm lẫn.** Những lớp mang tính nền tiếng ồn đô thị và tiếng máy móc như air_conditioner, engine_idling, drilling, jackhammer và siren có xu hướng bị nhầm lẫn lẫn nhau. Điều này thể hiện rõ qua các giá trị ngoài đường chéo chính trong confusion matrix, đặc biệt giữa engine_idling và jackhammer, cũng như giữa air_conditioner và street_music.
So sánh hai cấu hình cho thấy với AST-P, mức độ nhầm lẫn giữa các lớp tiếng ồn công nghiệp nhìn chung giảm so với AST-S, đặc biệt ở các lớp liên quan đến máy móc và động cơ. Điều này gợi ý rằng kiến thức học được từ AudioSet giúp backbone AST trích xuất đặc trưng âm thanh tốt hơn trong các bối cảnh phức tạp, dù vẫn còn tồn tại nhầm lẫn giữa các lớp có bản chất âm thanh rất gần nhau.

## 7.4 So sánh AST-S và AST-P

Từ kết quả trong Bảng 3, có thể rút ra các kết luận sau:

- Pretraining trên AudioSet giúp cải thiện đáng kể hiệu năng mô hình, từ 74.41% (AST-S) lên 80.08% (AST-P), tương đương mức tăng khoảng 5.7 điểm phần trăm trên UrbanSound8K.
- Độ lệch chuẩn giữa các fold của AST-P cao hơn AST-S, phản ánh sự khác biệt về độ khó giữa các fold, nhưng vẫn cho thấy xu hướng cải thiện nhất quán khi sử dụng pretraining AudioSet.
- So sánh giữa độ chính xác huấn luyện (khoảng 81–87%) và độ chính xác kiểm thử (khoảng 74–80%) cho thấy cả hai cấu hình đều tồn tại hiện tượng overfitting. Tuy nhiên, với cùng pipeline huấn luyện, AST-P thể hiện khả năng tổng quát hóa tốt hơn so với AST-S.

Nhìn chung, kết quả thực nghiệm trên UrbanSound8K tiếp tục khẳng định vai trò quan trọng của pretraining trên tập dữ liệu âm thanh quy mô lớn như AudioSet. Trong bối cảnh bài toán này, AST-P không chỉ đạt độ chính xác cao hơn mà còn cải thiện khả năng phân biệt các lớp âm thanh phức tạp, cho thấy lợi thế rõ rệt của pretraining đúng miền dữ liệu so với chỉ sử dụng pretraining trên ImageNet.

# 8 So sánh với bài báo gốc và thảo luận

## 8.1 So sánh định lượng với bài báo gốc

Do trong nghiên cứu gốc, tác giả không thực nghiệm trực tiếp trên tập dữ liệu UrbanSound8K nên phép so sánh được thực hiện gián tiếp thông qua việc đối chiếu **mức độ cải thiện hiệu năng** (performance gain) khi áp dụng pretraining trên AudioSet đối với các tập dữ liệu âm thanh môi trường có đặc tính tương đồng như ESC-50 và Speech Commands V2.

Kết quả thực nghiệm từ Bảng 3 cho thấy sự chuyển dịch từ cấu hình khởi tạo ImageNet (AST-S) sang ImageNet kết hợp AudioSet (AST-P) giúp độ chính xác trên UrbanSound8K tăng từ 74.41% lên 80.08%, tương đương mức cải thiện +5.67%. Mặc dù mức tăng này thấp hơn so với một số báo cáo trong bài báo gốc, xu hướng cải thiện vẫn hoàn toàn nhất quán với kết luận của tác giả AST. Cụ thể, trong nghiên cứu gốc [3], trên tập dữ liệu ESC-50 (có đặc tính gần với UrbanSound8K), việc sử dụng AudioSet pretraining giúp tăng độ chính xác từ 88.7% lên 95.6%, tương đương mức cải thiện +6.9%. Sự tương đồng về biên độ cải thiện cho thấy AudioSet đóng vai trò quan trọng trong việc cung cấp các biểu diễn âm thanh tổng quát mà mô hình khó có thể học được chỉ từ ImageNet. Những sai khác về mức độ cải thiện tuyệt đối có thể được lý giải bởi sự khác biệt về quy mô dữ liệu, cách chia fold, cấu hình huấn luyện và chiến lược regularization. Tuy nhiên, kết quả thực nghiệm vẫn khẳng định rằng AST không tự động học tốt các đặc trưng âm thanh chỉ từ dữ liệu hình ảnh, mà cần sự bổ trợ mạnh mẽ từ dữ liệu miền âm thanh như AudioSet để đạt hiệu năng tốt trên các bài toán phân loại âm thanh môi trường.

## 8.2 So sánh định lượng với mô hình CNN trên UrbanSound8K

So sánh trực tiếp trên cùng bộ dữ liệu UrbanSound8K, nhóm đối chiếu kết quả AST với mô hình CNN tiêu biểu trong bài báo của Salamon và Bello (2016) [7]. Trong nghiên cứu này, tác giả đề xuất kiến trúc SB-CNN và phân tích vai trò của data augmentation trong việc khắc phục hạn chế về quy mô dữ liệu gán nhãn cho bài toán phân loại âm thanh môi trường.

Theo phần *Results* của [7], mô hình SB-CNN khi huấn luyện không sử dụng augmentation đạt độ chính xác trung bình khoảng 0.73 (73%), tương đương với các baseline cùng thời điểm như SKM (0.74) và PiczakCNN (0.73). Khi kết hợp các kỹ thuật augmentation (tập "All" gồm Time

Stretching, Pitch Shifting, Dynamic Range Compression và Background Noise), độ chính xác của SB-CNN tăng lên khoảng 0.79 (79%), tương ứng với mức cải thiện xấp xỉ 6 điểm phần trăm. Kết quả này cho thấy trong thiết lập CNN thuần túy, augmentation đóng vai trò then chốt trong việc bù đắp sự thiếu hụt dữ liệu huấn luyện.

So với các mốc trên, kết quả của nhóm trong Bảng 3 cho thấy AST-S đạt độ chính xác trung bình 74.41%, thấp hơn SB-CNN khi sử dụng augmentation. Điều này phản ánh thực tế rằng việc chỉ khởi tạo từ ImageNet là chưa đủ để khai thác hiệu quả tập dữ liệu UrbanSound8K có quy mô và độ đa dạng hạn chế.

Ngược lại, cấu hình AST-P đạt độ chính xác trung bình 80.08%, vượt qua SB-CNN (augmented) khoảng 1 điểm phần trăm. Mặc dù mức chênh lệch không quá lớn, kết quả này đạt được mà không cần áp dụng các chiến lược augmentation mạnh như trong [7], cho thấy lợi thế của việc pretraining trên dữ liệu âm thanh quy mô lớn so với cách tiếp cận CNN phụ thuộc nhiều vào augmentation để nâng cao hiệu năng.

Ngoài ra, bài báo [7] sử dụng đầu vào log-Mel spectrogram với các TF-patch cố định dài 3 giây và tổng hợp dự đoán từ nhiều patch ở pha suy luận. Cách tiếp cận này khai thác hiệu quả các đặc trưng cục bộ theo thời gian và tần số, tuy nhiên vẫn bị giới hạn bởi khả năng học biểu diễn từ tập dữ liệu UrbanSound8K tương đối nhỏ. Ngược lại, AST-P được hưởng lợi từ các biểu diễn đã học trước trên AudioSet, giúp mô hình phân tách tốt hơn các lớp có đặc tính nhiễu nền và phổ tần chồng lấn mạnh như *air_conditioner*, *engine_idling*, *drilling* và *jackhammer*, phù hợp với các quan sát từ confusion matrix trong Bảng **??** và Bảng **??**.

Tóm lại, trong khi SB-CNN cho thấy hiệu quả của CNN kết hợp với data augmentation trong bối cảnh năm 2016, kết quả thực nghiệm của nhóm cho thấy hướng tiếp cận dựa trên Transformer kết hợp với pretraining trên dữ liệu âm thanh quy mô lớn mang lại hiệu năng cạnh tranh và ổn định hơn, ngay cả khi không phụ thuộc nhiều vào augmentation.

## 8.3 Điểm mạnh và điểm yếu của mô hình thực nghiệm

Dựa trên kết quả thực nghiệm và phân tích confusion matrix, các ưu nhược điểm của mô hình được xác định như sau:

**Điểm mạnh:**

- **Khả năng chuyển giao tri thức:** Mô hình cho thấy khả năng tận dụng hiệu quả tri thức từ

AudioSet, giúp đạt mức cải thiện rõ rệt về độ chính xác chỉ sau 3–5 epoch huấn luyện. Điều này cho thấy AST đã học được các biểu diễn âm thanh có tính tổng quát cao trong giai đoạn pretraining.

- **Khả năng phân biệt các lớp phức tạp:** So với AST-S, cấu hình AST-P giảm đáng kể mức độ nhầm lẫn giữa các lớp tiếng ồn công nghiệp và tiếng nền đô thị, thể hiện rõ qua các confusion matrix tổng hợp.

**Điểm yếu:**

- **Hiện tượng overfitting:** Độ chính xác trên tập huấn luyện (khoảng 81–87%) vẫn cao hơn đáng kể so với tập kiểm thử (khoảng 74–80%), cho thấy mô hình chưa hoàn toàn khắc phục được overfitting khi làm việc với tập dữ liệu quy mô vừa và nhỏ như UrbanSound8K.
- **Khó khăn với các lớp nhiễu nền:** Các lớp như *air_conditioner*, *engine_idling* và *drilling* vẫn còn bị nhầm lẫn tương đối nhiều do đặc trưng phổ tần chồng lấn trong biểu diễn log-Mel spectrogram.

## 8.4 Đề xuất cải tiến và hướng phát triển

Để tiếp tục cải thiện hiệu năng và thu hẹp khoảng cách với các kết quả SOTA được báo cáo trong bài báo gốc, một số hướng cải tiến được đề xuất:

1. **Tăng cường chiến lược Augmentation:** Áp dụng mạnh hơn các kỹ thuật như SpecAugment hoặc Mixup với hệ số phù hợp có thể giúp giảm overfitting và cải thiện khả năng tổng quát hóa trên UrbanSound8K.
2. **Sử dụng Model Ensemble:** Kết hợp nhiều checkpoint từ các epoch khác nhau hoặc các fold khác nhau được kỳ vọng sẽ cải thiện độ ổn định và độ chính xác tổng thể, tương tự chiến lược ensemble được sử dụng trong [3].
3. **Tối ưu hóa độ phân giải thời gian–tần số:** Giảm stride theo trục thời gian hoặc tần số có thể giúp mô hình nắm bắt tốt hơn các đặc trưng ngắn hạn của các lớp khó như *drilling* và *jackhammer*, mặc dù sẽ làm tăng chi phí tính toán.

# 9 Mô tả ứng dụng

Nhóm đã xây dựng một ứng dụng web hoàn chỉnh để phân tích và phân loại âm thanh đô thị (UrbanSound8K). Ứng dụng không chỉ đưa ra dự đoán nhãn (như tiếng còi xe, tiếng nhạc đường phố...) mà còn cung cấp cái nhìn sâu sâu bên trong dữ liệu thông qua các biểu đồ trực quan hóa. Hệ thống cho phép người dùng tải file âm thanh hoặc ghi âm trực tiếp, sau đó xử lý qua mô hình Audio Spectrogram Transformer (AST).
Điểm nổi bật của phiên bản này là khả năng hiển thị biểu đồ **Log-Mel Spectrogram** và biểu đồ xác suất cho từng lớp, giúp người dùng hiểu rõ hơn về độ tin cậy của mô hình.

Các công nghệ và thư viện chính:

- **Streamlit:** Xây dựng giao diện tương tác, xử lý luồng nhập liệu từ microphone và file upload.
- **PyTorch & Torchaudio:** Nền tảng Deep Learning dùng để vận hành mô hình AST và chuyển đổi tín hiệu âm thanh sang dạng phổ.
- **Librosa & Matplotlib:** Được bổ sung để vẽ biểu đồ trực quan. *Librosa* hỗ trợ hiển thị Spectrogram với thang đo Mel chuẩn xác, trong khi *Matplotlib* vẽ biểu đồ cột phân phối xác suất với màu sắc trực quan.
- **Soundfile & Numpy:** Xử lý đọc/ghi file âm thanh và các phép toán ma trận số học.

## 9.1 Cơ chế Ensemble Learning

Để tăng độ chính xác và giảm thiểu variance của mô hình, ứng dụng áp dụng kỹ thuật **Ensemble Averaging** trên 10 folds của bộ dữ liệu UrbanSound8K. Thay vì chỉ dựa vào một bộ trọng số duy nhất, hệ thống thực hiện quy trình sau:

1. **Load Multi-Weights:** Hệ thống lần lượt tải trọng số từ 10 file model đã được huấn luyện trước (từ `ast_us8k_fold1.pth` đến `ast_us8k_fold10.pth`).
2. **Softmax Averaging:** Với mỗi model $i$ (trong 10 models), input được đưa qua mạng để tính ra vector xác suất $P_i$ thông qua hàm Softmax:

$$P_i = \text{Softmax}(\text{Model}_i(x))$$

3. **Tổng hợp kết quả:** Kết quả cuối cùng là trung bình cộng của các vector xác suất từ tất cả các models hợp lệ:

$$P_{\text{final}} = \frac{1}{N} \sum_{i=1}^{N} P_i$$

4. **Quyết định nhãn:** Nhãn có giá trị xác suất cao nhất trong vector $P_{\text{final}}$ sẽ được chọn làm kết quả dự đoán (Argmax).

Phương pháp này giúp mô hình hoạt động ổn định hơn, tránh việc bị nhiễu bởi các đặc điểm cục bộ (bias) của một lần huấn luyện đơn lẻ.

## 9.2 Hướng dẫn sử dụng

- **Bước 1 - Cấu hình:** Tại thanh bên trái, chọn phiên bản kiến trúc mô hình (AST-P hoặc AST-S). Hệ thống sẽ hiển thị đường dẫn và mô tả tương ứng.
- **Bước 2 - Nhập liệu:** Người dùng chọn tab **"Upload File"** để tải file .wav hoặc tab **"Ghi âm"** để thu tín hiệu trực tiếp.
- **Bước 3 - Phân tích:** Nhấn nút **"Chạy mô hình..."**. Hệ thống sẽ hiển thị thanh tiến trình chạy qua 10 folds.
- **Bước 4 - Xem kết quả:**
  - **Kết quả chính:** Hiển thị nhãn dự đoán và độ tin cậy tổng hợp.
  - **Audio Player:** Nghe lại đoạn âm thanh vừa phân tích.
  - **Mel Spectrogram:** Biểu đồ nhiệt (heatmap) thể hiện đặc trưng âm thanh đầu vào.
  - **Xác suất các lớp:** Biểu đồ cột ngang so sánh tỉ lệ dự đoán giữa 10 lớp, thanh màu xanh lá cây biểu thị kết quả được chọn.

# 10 Kết luận

Nghiên cứu này đã triển khai thành công mô hình Audio Spectrogram Transformer (AST) cho bài toán phân loại âm thanh môi trường trên tập dữ liệu UrbanSound8K, đồng thời kiểm chứng hiệu quả của việc chuyển giao tri thức đa miền. Kết quả thực nghiệm cho thấy cấu hình sử dụng trọng số tiền huấn luyện từ AudioSet (AST-P) đạt độ chính xác trung bình 80.08%, vượt trội so với cấu hình chỉ khởi tạo từ ImageNet (74.41%). Mức chênh lệch khoảng 5.7% này củng cố mạnh mẽ giả thuyết của nghiên cứu gốc [3]: đối với các tập dữ liệu âm thanh quy mô vừa và nhỏ, việc thừa hưởng các đặc trưng thính giác được học từ tập dữ liệu lớn là yếu tố then chốt để mô hình Transformer hội tụ tốt và đạt hiệu năng cao, thay vì chỉ dựa vào các đặc trưng thị giác từ miền hình ảnh.

# 11 Tài liệu tham khảo

[1] Shawn Hershey et al. "CNN architectures for large-scale audio classification". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, pp. 131–135. DOI: 10.1109/ICASSP.2017.7952132.

[2] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *arXiv preprint arXiv:2010.11929* (2020). URL: https://arxiv.org/abs/2010.11929.

[3] Yuan Gong, Yu-An Chung, and James Glass. "AST: Audio Spectrogram Transformer". In: *arXiv preprint arXiv:2104.01778* (2021). URL: https://arxiv.org/abs/2104.01778.

[4] C.-H. Lee, C.-H. Lin, and B.-H. Juang. "A study on speaker adaptation of the parameters of continuous density hidden Markov models". In: *IEEE Transactions on Signal Processing* 39.4 (1991), pp. 806–814. DOI: 10.1109/78.80902.

[5] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 25. Curran Associates, Inc., 2012, pp. 1097–1105. URL: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

[6] YuanGongND. *GitHub - YuanGongND/ast: Code for the Interspeech 2021 paper "AST: Audio Spectrogram Transformer"*. en. URL: https://github.com/YuanGongND/ast/tree/master.

[7] Justin Salamon and Juan Pablo Bello. *Deep Convolutional Neural Networks and Data Augmentation for Environmental Sound Classification*. 2016. arXiv: 1608.04363 [cs.SD].