**Báo cáo nội dung về 2 framework nhận diện giọng nói HTK và Sphinx 4**

# **2.** **Tổng quan về HTK (Hidden Markov Model Toolkit)**

## 2.1. Giới thiệu HTK

**HTK (Hidden Markov Model Toolkit)** là một bộ công cụ phần mềm được phát triển bởi **Cambridge University Engineering Department** nhằm phục vụ mục đích nghiên cứu và phát triển các hệ thống nhận dạng mẫu, đặc biệt là **nhận diện giọng nói**.

**HTK** là một bộ công cụ phần mềm tích hợp dùng để xây dựng và xử lý các mô hình Markov ẩn liên tục (continuous density HMMs). Được viết bằng ngôn ngữ lập trình ANSI C, HTK chủ yếu chạy trên các hệ điều hành UNIX nhưng cũng tương thích với các hệ điều hành hiện đại khác.

Mặc dù mang tính tổng quát, HTK vẫn khuyến khích một cách tiếp cận nhất định trong phát triển hệ thống nhận dạng tiếng nói:

1. **Ưu tiên mô hình liên tục**: HTK hỗ trợ mô hình liên tục thay vì rời rạc vì tính ổn định và chính xác cao hơn trong thực tiễn và nghiên cứu.
2. **Hỗ trợ ràng buộc tham số (parameter tying)**: Cho phép chia sẻ tham số giữa các mô hình để tăng hiệu quả và giảm độ phức tạp.
3. **Tiếp cận phát triển mô hình từng bước**: Hệ thống được tinh chỉnh qua nhiều giai đoạn, xen kẽ giữa việc chỉnh sửa mô hình và tái huấn luyện.
4. **Cung cấp công cụ tích hợp mạnh mẽ**: Hỗ trợ xử lý các loại dữ liệu khác nhau như âm thanh, bản phiên âm và từ điển.

Công cụ này được phát triển bởi **Steve Young** và nhóm nghiên cứu tại nhóm **Speech Vision and Robotics** của Đại học **Cambridge**, với nhiều thư viện và module khác nhau phục vụ các chức năng riêng biệt.

## 2.2. Kiến trúc hệ thống HTK

HTK là một hệ thống dòng lệnh, bao gồm nhiều công cụ nhỏ thực hiện các chức năng riêng biệt trong toàn bộ pipeline nhận diện giọng nói. Kiến trúc HTK được chia thành ba giai đoạn chính:

* **Tiền xử lý âm thanh**: chuyển đổi tín hiệu âm thanh thô thành các đặc trưng số học.
* **Huấn luyện mô hình HMM**: sử dụng dữ liệu huấn luyện để xây dựng mô hình.
* **Nhận dạng**: so sánh đầu vào mới với các mô hình đã huấn luyện để xác định kết quả.

## 2.3. Quy trình nhận diện giọng nói trong HTK

Quy trình hoạt động của HTK thường bao gồm các bước sau:

1. **Thu thập dữ liệu âm thanh** (file .wav)
2. **Trích xuất đặc trưng âm thanh** bằng công cụ HCopy (chuyển .wav sang .mfc - Mel Frequency Cepstral Coefficients)
3. **Tính toán trung bình đặc trưng** với HCompV
4. **Huấn luyện mô hình HMM** với các bước lặp HERest (huấn luyện Baum-Welch)
5. **Tạo mạng ngôn ngữ đơn giản** (word network hoặc grammar)
6. **Nhận diện giọng nói** với HVite

## 2.4. Các thành phần chính (HCopy, HCompV, HERest, HVite, v.v.)

|  |  |
| --- | --- |
| **Công cụ** | **Chức năng** |
| HCopy | Trích xuất đặc trưng MFCC từ âm thanh |
| HCompV | Tính toán trung bình và khởi tạo HMM |
| HERest | Huấn luyện mô hình HMM |
| HParse | Chuyển đổi văn phạm sang word network |
| HVite | Thực hiện nhận dạng giọng nói |
| HResults | Đánh giá kết quả nhận dạng so với ground truth |

## 2.5. Ưu điểm và hạn chế của HTK

**Ưu điểm:**

* Khả năng kiểm soát chi tiết pipeline nhận diện
* Hỗ trợ linh hoạt nhiều loại dữ liệu và mô hình HMM
* Dùng phổ biến trong nghiên cứu, đặc biệt ở lĩnh vực học thuật
* Tài liệu hướng dẫn khá đầy đủ

**Hạn chế:**

* Khó sử dụng cho người mới do giao diện dòng lệnh và cấu hình phức tạp
* Thiếu GUI trực quan
* Không còn được cập nhật thường xuyên
* Hạn chế khi áp dụng trong các hệ thống thời gian thực hoặc quy mô lớn

## 2.6. Ứng dụng thực tế của HTK

HTK đã từng được sử dụng trong:

* Các nghiên cứu học thuật về nhận dạng giọng nói
* Các hệ thống nhận dạng lệnh bằng giọng nói đơn giản
* Dự án phát triển hệ thống TTS (Text-to-Speech) và ASR (Automatic Speech Recognition)
* Là cơ sở nền tảng cho các công cụ như Julius (ASR cho tiếng Nhật)

## 2.7. Quá trình thực hiện và thao tác với HTK

Trong phần này, chúng tôi sẽ trình bày các bước thao tác thực tế khi sử dụng HTK để xây dựng một hệ thống nhận diện giọng nói đơn giản. Các bước sẽ bao gồm: chuẩn bị dữ liệu, trích xuất đặc trưng, huấn luyện mô hình và nhận dạng.

**📁 Bước 1: Chuẩn bị dữ liệu âm thanh**

* Tạo thư mục lưu file .wav
* Ghi âm các câu lệnh cơ bản bằng giọng người

bash

Sao chépChỉnh sửa

mkdir wav\_data

# chèn hình ảnh minh họa thư mục wav\_data chứa file .wav tại đây

**🧪 Bước 2: Trích xuất đặc trưng MFCC**

Sử dụng HCopy để chuyển đổi file .wav sang định dạng .mfc chứa đặc trưng âm thanh.

bash

Sao chépChỉnh sửa

HCopy -T 1 -C config/hcopy.cfg -S wav2mfc.scp

# chèn ảnh minh họa file .mfc được sinh ra

*Gợi ý*: Tập tin wav2mfc.scp chứa danh sách các đường dẫn file .wav và .mfc.

**🧠 Bước 3: Khởi tạo HMM**

Dùng HCompV để tính toán giá trị trung bình và tạo mô hình HMM cơ bản.

bash

Sao chépChỉnh sửa

HCompV -C config/train.cfg -f 0.01 -m -S train.scp -M hmm0 proto

# chèn ảnh: mô hình HMM ban đầu (folder hmm0)

**🔁 Bước 4: Huấn luyện HMM**

Huấn luyện mô hình HMM qua nhiều vòng lặp bằng HERest:

bash

Sao chépChỉnh sửa

HERest -C config/train.cfg -I labels.mlf -t 250.0 150.0 1000.0 -S train.scp -H hmm0/macros -H hmm0/hmmdefs -M hmm1 monophones0

# chèn ảnh: log file huấn luyện hoặc mô hình hmm1

*Ghi chú*: Bạn cần định nghĩa các nhãn phoneme trong monophones0 và labels.mlf.

**🧩 Bước 5: Tạo mạng ngôn ngữ đơn giản**

Tạo cấu trúc ngôn ngữ sử dụng HParse từ file ngữ pháp định nghĩa:

bash

Sao chépChỉnh sửa

HParse grammar.txt wdnet

# chèn ảnh: sơ đồ mạng từ (word network)

**🔊 Bước 6: Nhận diện bằng HVite**

Thực hiện nhận dạng file âm thanh đầu vào:

bash

Sao chépChỉnh sửa

HVite -H hmm1/macros -H hmm1/hmmdefs -S test.scp -i result.mlf -w wdnet -p 0.0 -s 5.0 dict monophones0

# chèn ảnh: kết quả nhận dạng hiển thị trong file result.mlf

**📊 Bước 7: Đánh giá kết quả**

So sánh kết quả nhận dạng với nhãn thực bằng HResults:

bash

Sao chépChỉnh sửa

HResults -I ref.mlf monophones0 result.mlf

# chèn ảnh: bảng kết quả đánh giá độ chính xác

📌 **Lưu ý**: Để chạy được các lệnh trên, bạn cần chuẩn bị đầy đủ các file cấu hình: config/\*.cfg, danh sách phoneme, từ điển (dict), danh sách tệp huấn luyện (train.scp), v.v.