**Báo cáo Bài tập lớn môn Tạo sinh âm thanh**

**Đề tài: Automatic Speech Recognition**

1. **Giới thiệu bài toán**

Với sự phát triển mạnh mẽ của khoa học máy tính và trí tuệ nhân tạo, việc ứng dụng các công nghệ tiên tiến vào đời sống hàng ngày đang trở nên ngày càng phổ biến. Dữ liệu ngày nay không chỉ giới hạn ở dạng số mà còn rất đa dạng, từ hình ảnh, video cho đến âm thanh. Trong số đó, dữ liệu âm thanh đóng vai trò quan trọng trong việc xây dựng các hệ thống tương tác giữa con người và máy tính.

Một trong những ứng dụng nổi bật của dữ liệu âm thanh là bài toán **Nhận dạng giọng nói tự động** – *Automatic Speech Recognition* (ASR). Đây là công nghệ cho phép máy tính tự động chuyển đổi tín hiệu âm thanh lời nói thành văn bản. ASR đóng vai trò như một cầu nối giữa giao tiếp tự nhiên của con người (nói) và khả năng hiểu/ngôn ngữ của máy tính. Mục tiêu của ASR là chuyển đổi chuỗi tín hiệu âm thanh thành câu văn có ngữ pháp và ngữ nghĩa chính xác nhất có thể.

Trong đề tài này, nhóm chúng tôi gồm 4 thành viên lựa chọn nghiên cứu và triển khai mô hình **Wav2Vec 2.0 kết hợp với CTC (Connectionist Temporal Classification)** – một trong những mô hình ASR tiên tiến được phát triển bởi Facebook AI (nay là Meta AI) vào năm 2020. Wav2Vec 2.0 là mô hình học tự giám sát (self-supervised learning), có khả năng trích xuất đặc trưng từ tín hiệu âm thanh thô (raw audio) và huấn luyện hiệu quả ngay cả khi lượng dữ liệu được gán nhãn là hạn chế.

Mục tiêu của nhóm là áp dụng mô hình Wav2Vec 2.0 cho ngôn ngữ tiếng Việt nhằm xây dựng một hệ thống nhận diện giọng nói phù hợp với đặc thù ngôn ngữ này.

Cụ thể, nhóm đã sử dụng bộ dữ liệu gồm **56.427 cặp audio (.wav) và transcript (.txt)**. Quá trình huấn luyện được thực hiện trên nền tảng Kaggle, sử dụng GPU P100 để đảm bảo hiệu năng xử lý. Mô hình được đánh giá trên hai tập dữ liệu: *public test* và *private test*, với chỉ số **Word Error Rate (WER)** được đo lường thông qua nền tảng AIHub để đảm bảo tính khách quan và chính xác.

1. **Tiền xử lý dữ liệu huấn luyện**
2. **Lọc dữ liệu và phân chia dữ liệu**

Thống kê trên toàn bộ tập dữ liệu huấn luyện có 56 472 cặp dữ liệu gồm một file âm thanh wav và một file transcription txt, trong đó có:

* 689 cặp có file âm thanh dưới 1s
* 8352 cặp có file âm thanh trên 10s
* 47 386 cặp có file âm thanh trên 1s và dưới 10s

Tiến hành loại 689 cặp dữ liệu có file âm thanh dưới 1s. Còn lại chia làm 2 nhóm để xử lý riêng với từng mô hình ASR khác nhau.

Đối với các file transcript, tiến hành kiểm tra xem có xuất hiện ký tự đặc biệt, khoảng trắng thừa và loại bỏ chúng.

1. **Tăng cường dữ liệu**

Trong quá trình tăng cường dữ liệu cho huấn luyện mô hình ASR, áp dụng ba kỹ thuật chính: thêm nhiễu trắng, time warping, và kết hợp cả hai.

Nhiễu trắng được thêm vào tín hiệu âm thanh với biên độ ngẫu nhiên nhằm mô phỏng các điều kiện nhiễu nền thực tế như tiếng gió hoặc tiếng quạt. Kỹ thuật này giúp mô hình học cách nhận dạng tiếng nói trong môi trường không lý tưởng.

Time warping là kỹ thuật biến đổi trục thời gian của tín hiệu âm thanh với tỷ lệ ngẫu nhiên, giúp mô phỏng các tốc độ nói khác nhau và tăng khả năng thích nghi với giọng nói đa dạng.

Trong một số trường hợp, chúng tôi kết hợp đồng thời cả hai kỹ thuật để tạo ra dữ liệu đầu vào vừa bị nhiễu vừa bị biến dạng thời gian, từ đó làm tăng tính đa dạng và độ bền vững của mô hình khi triển khai thực tế..

1. **Lọc nhiễu trên các file âm thanh**

**3.1. Chuẩn hóa âm thanh**

Đảm bảo tất cả file âm thanh có biên độ đồng đều. Sử dụng thư viện librosa để tải file âm thanh với tần số mẫu 16kHz (phù hợp cho ASR) và chuyển sang kênh đơn (mono). Biên độ được chuẩn hóa bằng cách chia tín hiệu cho giá trị tuyệt đối lớn nhất

**3.2. Lọc nhiễu nền bằng Pectral Subtraction**

Phương pháp này là phương pháp truyền thống, không sử dụng học sâu. Phương pháp này thực hiện khử nhiễu bằng cách chuyển tín hiệu tiếng nói sang miền tần số thông qua biến đổi Fourier ngắn hạn (STFT). Ước lượng nhiễu từ vài khung đầu tiên, sau đó trừ nhiễu khỏi tín hiệu gốc, rồi khôi phục lại âm thanh sạch.

**3.3 Giảm vang (Dereverberation)**

Sử dụng công cụ VoiceFixer để xử lý tín hiệu âm thanh. VoiceFixer là một mô hình dựa trên học sâu (deep learning), được huấn luyện để khôi phục chất lượng âm thanh bằng cách giảm độ vang. Tín hiệu âm thanh sau các bước tiền xử lý trước được đưa vào VoiceFixer ở chế độ giảm vang (dereverb). Mô hình phân tích và tái tạo tín hiệu để loại bỏ các thành phần vang, giữ lại giọng nói chính.

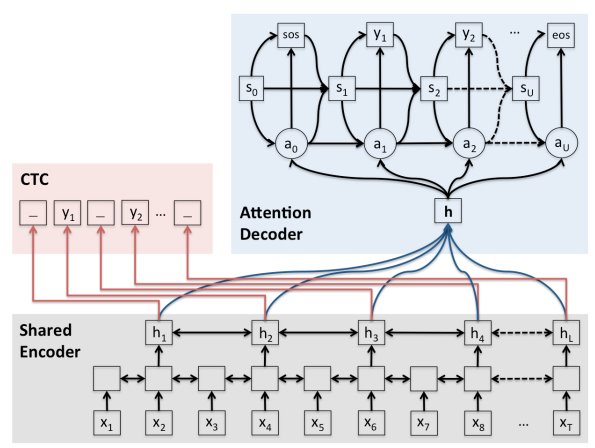
**Phương pháp**: VoiceFixer sử dụng mạng nơ-ron sâu (deep neural network) để ánh xạ tín hiệu âm thanh có vang sang tín hiệu sạch. Không giống các phương pháp truyền thống (như lọc tuyến tính hoặc xử lý phổ), VoiceFixer học các đặc trưng phức tạp của âm thanh từ dữ liệu huấn luyện, giúp xử lý hiệu quả hơn trong các môi trường có độ vang đa dạng.

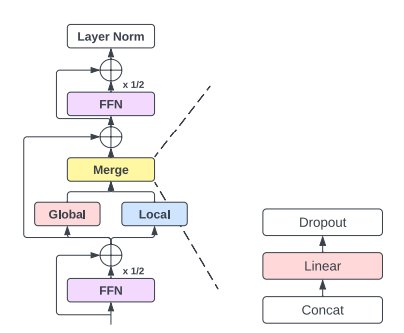
1. **Hướng tiếp cận**
2. **Cấu trúc model sử dụng:**
   1. **Dolphin**
   2. **Wav2vec**
   3. **Whiper**

**Tổng quan mô hình:**

Dolphin là một mô hình nhận dạng giọng nói tự động (ASR) đa ngôn ngữ được thiết kế đặc biệt để cải thiện hiệu suất cho các ngôn ngữ Đông Á, Nam Á, Đông Nam Á và Trung Đông trong đó có tiếng Việt.

Dựa trên phương pháp được trình bày trong OWSM, mô hình áp dụng kiến trúc kết hợp CTC-Attention, giúp tận dụng được ưu điểm của cả hai cơ chế: Connectionist Temporal Classification (CTC) và Attention. Cách tiếp cận lai này cho phép huấn luyện mô hình nhận dạng tiếng nói đa ngôn ngữ trên quy mô lớn một cách hiệu quả và ổn định.

Khung mô hình end-to-end kết hợp CTC-attention: bộ mã hóa dùng chung được huấn luyện đồng thời với hai mục tiêu của mô hình CTC và mô hình attention. Bộ mã hóa này chuyển đổi chuỗi đầu vào x thành các đặc trưng cấp cao h, sau đó bộ giải mã attention dựa trên vị trí (location-based attention decoder) sẽ sinh ra chuỗi ký tự y

**Bộ mã hóa (encoder) trong kiến trúc của chúng tôi dựa trên E-Branchformer — một mô hình tiên tiến tích hợp các nhánh xử lý song song. Thiết kế này giúp mô hình nắm bắt hiệu quả cả các phụ thuộc cục bộ và toàn cục trong tín hiệu giọng nói đầu vào. Về phía bộ giải mã (decoder), chúng tôi sử dụng Transformer tiêu chuẩn, vốn đã chứng minh được hiệu quả trong các bài toán chuyển đổi chuỗi** 

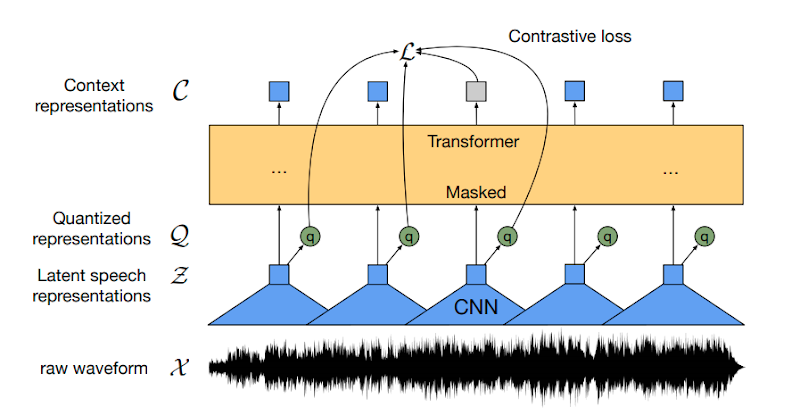
E-Branchformer block với merge model dung trong Branchformer

**Để nâng cao hiệu quả huấn luyện và hiệu suất xử lý, mô hình tích hợp 4× lớp giảm mẫu (subsampling), giúp rút ngắn độ dài chuỗi đặc trưng đầu vào và tăng tốc độ tính toán. Mô hình huấn luyện bốn phiên bản mô hình tương ứng với các kích thước của Whisper: base, small, medium và large. Ở mỗi mức độ, mô hình Dolphin có số lượng tham số nhỉnh hơn Whisper một chút, do sử dụng bộ mã hóa E-Branchformer, lớp CTC và lớp giảm mẫu.**

Trong nội dung môn học, nhóm đã sử dụng model dolphin-small

**Mô hình Wav2Vec :**

**Wav2Vec 2.0** là một mô hình học tự giám sát (self-supervised learning) dành cho xử lý tín hiệu âm thanh, được giới thiệu bởi Facebook AI (nay là Meta AI) vào năm 2020. Mô hình này đã tạo ra bước đột phá trong lĩnh vực **nhận dạng giọng nói tự động (ASR)** nhờ khả năng học đặc trưng âm thanh từ dữ liệu thô (raw audio) mà không cần gán nhãn, giúp giảm mạnh chi phí và công sức gán nhãn dữ liệu huấn luyện.



Mô hình được xây dựng một bộ mã hóa đặc trưng tích chập đa lớp : f : X → Z với đầu vào là âm thanh thô và xuất ra đại diện các ký tự được quy định bởi vocab z1 , …. , zT cho trước T thời gian. Sau đó được qua lớp Transformer để g : Z → C để xây dựng biểu diễn c1 , … , cT  để nắm bắt thông tin từ chuỗi. Đầu ra của bộ mã hóa đặc trưng được phân loại thành qt bằng một mô-đun định lượng Z → Q để biểu diễn các mục tiêu trong mục tiêu tự giám sát. So với vq-wav2vec , mô hình Wav2Vec2 xây dựng các biểu diễn ngữ cảnh trên các biểu diễn âm thanh liên tục và cơ chế tự chú ý (self-attention) nắm bắt sự phụ thuộc trong toàn bộ chuỗi các biểu diễn ẩn theo cách đầu cuối.

Wav2Vec 2.0 gồm **ba thành phần chính**:

#### **Feature Encoder (Bộ mã hóa đặc trưng)**

Là một chuỗi các lớp **convolution 1D** hoạt động trên dạng sóng âm thanh đầu vào (raw waveform).

Mục tiêu: Trích xuất các đặc trưng cục bộ từ tín hiệu âm thanh, chuyển từ chuỗi tín hiệu thô sang biểu diễn đặc trưng có kích thước thấp hơn.

#### **Context Network (Mạng học ngữ cảnh)**

Là một **transformer encoder** sâu, gồm nhiều lớp self-attention (giống trong kiến trúc BERT hoặc GPT).

Mục tiêu: Mô hình hóa thông tin dài hạn và mối quan hệ ngữ cảnh giữa các khung thời gian trong âm thanh.

Kết quả là các vector biểu diễn có ngữ cảnh đầy đủ, dùng cho cả huấn luyện và suy luận.

#### **Quantization Module (Bộ lượng tử hóa)**

Tạo ra các mã ẩn (latent discrete units) từ đầu ra của feature encoder bằng cách ánh xạ sang một không gian rời rạc.

Mục tiêu: Tạo mục tiêu học cho nhiệm vụ pretraining bằng cách so sánh đầu ra context vector với các mã lượng tử hóa này.

Đây là phần làm nên “self-supervised learning” của mô hình: mô hình học cách dự đoán các mã rời rạc trong tương lai từ bối cảnh hiện tại.

Đối với việc đào tạo tự giám sát, chúng tôi phân biệt đầu ra của bộ mã hóa đặc trưng **z** thành một tập hợp hữu hạn các đại diện giọng nói thông qua lượng tử hóa theo sản phẩm. Lựa chọn này đã dẫn đến những kết quả tốt trong các công trình trước đó, nơi mà các đơn vị rời rạc được học trong bước đầu tiên, tiếp theo là học các đại diện theo ngữ cảnh. Lượng tử hóa theo sản phẩm tương đương với việc chọn các đại diện đã lượng tử hóa từ nhiều sách mã và nối chúng lại một cách tuần tự. Với *G* sách mã, hoặc các nhóm, có *V* mục *e* RV\*d/G, chúng tôi chọn một mục từ mỗi sách mã và nối các vector thu được e1,…. , eG lại với nhau và áp dụng một phép biến đổi tuyến tính Rd → Rf để thu được **q**  Rf.

Gumbel softmax cho phép chọn các mục trong bảng mã rời rạc theo cách hoàn toàn có thể phân biệt. Chúng tôi sử dụng bộ ước lượng đi thẳng và thiết lập *G* phép toán Gumbel softmax khó. Đầu ra mã hóa tính năng *z* được ánh xạ tới l  RGxV logits và xác suất để chọn mục bảng mã thứ *v* cho nhóm *g* là



Ở  giá trị không âm, n = - log( -log( u ) ) và *u* là các mẫu đồng nhất từ U **(0, 1)**. Trong bước đi tới, mã code *i* được chọn bằng *i* = argmaxjpg,jvà trong bước lùi, gradient thực sự của các đầu ra Gumbel softmax được sử dụng.

**Training model**

Để tiền huấn luyện mô hình, chúng tôi che khuất một tỉ lệ nhất định các bước thời gian trong không gian mã hóa đặc trưng tiềm ẩn, tương tự như mô hình ngôn ngữ bị che dấu trong BERT. Mục tiêu đào tạo yêu cầu xác định đại diện âm thanh tiềm ẩn đã được định lượng đúng trong một tập hợp các yếu tố phân tâm cho mỗi bước thời gian bị che dấu và mô hình cuối cùng được tinh chỉnh trên dữ liệu đã được gán nhãn

Chúng tôi che khuất một tỉ lệ của các đầu ra mã hóa đặc trưng, hoặc các bước thời gian trước khi cho chúng vào mạng ngữ cảnh và thay thế chúng bằng một vectơ đặc trưng đã được đào tạo chia sẻ giữa tất cả các bước thời gian bị che khuất; chúng tôi không che khuất các đầu vào cho mô-đun lượng tử hóa. Để che khuất các đại diện âm thanh tiềm ẩn được đầu ra bởi bộ mã hóa, chúng tôi chọn ngẫu nhiên không thay thế một tỉ lệ nhất định p của tất cả các bước thời gian để làm các chỉ số bắt đầu và sau đó che khuất M bước thời gian liên tiếp tiếp theo từ mỗi chỉ số được chọn; các khoảng thời gian có thể chồng chéo.

Trong giai đoạn tiền huấn luyện, chúng tôi học các biểu diễn của âm thanh nói bằng cách giải quyết một nhiệm vụ tương phản *L*m, yêu cầu xác định đại diện âm thanh nói được mã hóa thực sự cho một bước thời gian bị che khuất trong tập hợp các yếu tố gây nhiễu. Điều này được bổ sung bởi một mất mát đa dạng mã thông báo *L*d để khuyến khích mô hình sử dụng các mục mã thông báo một cách bình đẳng.



Với là một tham số điều chỉnh

Hàm mất mát tương phản :

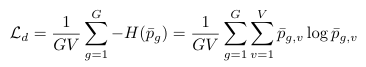
Dựa trên đầu ra mạng ngữ cảnh ct được căn chỉnh tại thời điểm bị che khuất t, mô hình cần xác định đại diện âm thanh tiềm ẩn lượng tử thực sự q*t* trong một tập hợp các đại diện ứng viên lượng tử q  Qt gồm K + 1 đại diện, bao gồm q*t* và K đại diện gây nhầm lẫn. Các đại diện gây nhầm lẫn được lấy mẫu đồng nhất từ các thời điểm bị che khuất khác của cùng một phát biểu. Hàm mất mát được định nghĩa là



nơi chúng tôi tính toán độ tương đồng cosine sim(a, b) = aTb / ||a||.||b|| giữa các đại diện ngữ cảnh và các đại diện giọng nói tiềm ẩn đã định lượng.

Hàm mất mát đa dạng :

Nhiệm vụ tương phản phụ thuộc vào mã sách để đại diện cho cả ví dụ tích cực và tiêu cực và mất mát đa dạng *L*d được thiết kế để tăng cường việc sử dụng các đại diện mã hóa đã định lượng. Chúng tôi khuyến khích việc sử dụng bình đẳng các mục V trong mỗi G mã sách bằng cách tối đa hóa entropy của phân phối softmax trung bình l trên các mục mã sách cho mỗi 3 mã sách pg qua một lô câu nói; phân phối softmax không chứa tiếng ồn gumbel cũng như nhiệt độ.

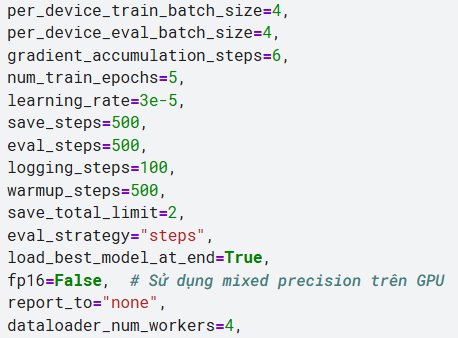


Fine-tune mô hình nhận dạng giọng nói bằng cách thêm một phép chiếu tuyến tính được khởi tạo ngẫu nhiên lên trên mạng ngữ cảnh thành C lớp đại diện cho từ vựng của nhiệm vụ. Đối với Librispeech, chúng tôi có 29 token cho các mục tiêu ký tự cộng với một token ranh giới từ. Các mô hình được tối ưu hóa bằng cách giảm thiểu tổn thất CTC và chúng tôi áp dụng một phiên bản sửa đổi của SpecAugment bằng cách che khuất các bước thời gian và kênh trong quá trình huấn luyện, điều này làm chậm sự quá khớp và cải thiện đáng kể tỷ lệ lỗi cuối cùng, đặc biệt là trên các tập hợp Libri-light với ít ví dụ được gán nhãn.

Tiến hành thực nghiệm :

Thiết lập môi trường training ở kaggle với sử dụng GPU P100. Tiến hành training với bộ dữ liệu tiếng Việt gồm hơn 56000 bản audio (.Wav) và bản text tương ứng (.txt) .

Thông số training :



**Quá trình kiểm thử :**

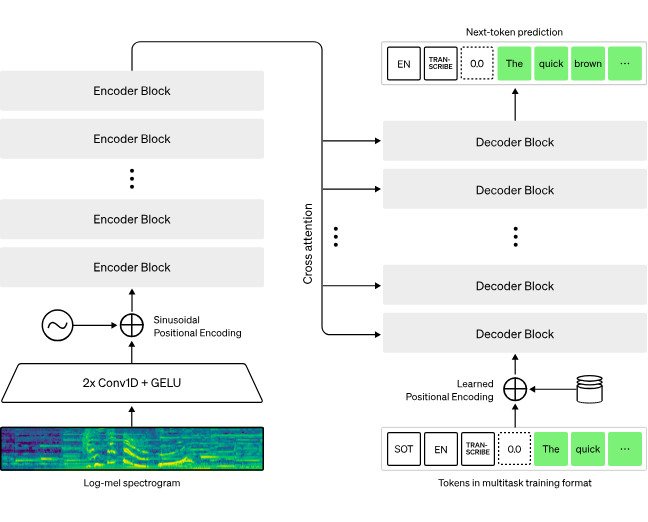
Để kiểm thử mô hình chúng tôi sử dụng 2 tập dữ liệu được cung cấp là public test và private test

Kết quả kiểm thử được được đánh giá qua tỷ lệ từ lỗi Word error rate trên nền tảng AIHuB

**b. Mô hình Whisper :**

Whisper là một mô hình nhận dạng giọng nói tự động (Automatic Speech Recognition ) được phát triển bởi OpenAI, với mã nguồn được công bố rộng rãi. Mô hình này được huấn luyện trên một tập dữ liệu lớn bao gồm khoảng 680.000 giờ âm thanh đa ngôn ngữ và đa tác vụ, giúp cải thiện độ chính xác trong việc chuyển đổi giọng nói thành văn bản trong nhiều ngữ cảnh khác nhau.

Whisper hỗ trợ nhận diện hơn 90 ngôn ngữ và có khả năng tự động phát hiện ngôn ngữ đầu vào. Ngoài ra, mô hình còn tích hợp chức năng dịch giọng nói sang tiếng Anh và thực hiện các tác vụ như phân đoạn lời thoại, gán thời gian và xử lý tiếng ồn ở mức độ cơ bản.



Mô hình được xây dựng theo kiến trúc Transformer encoder-decoder. Tất cả các tệp âm thanh được chuẩn hóa về tần số 16kHz, chuyển thành Mel spectrogram (80 kênh, cửa sổ 25ms, bước nhảy 10ms). Dữ liệu được chuẩn hóa trong khoảng -1 đến 1 và có giá trị trung bình gần bằng 0 trên toàn bộ tập pretrain.

Phần encoder bắt đầu bằng hai lớp convolution (bộ lọc rộng 3, hàm kích hoạt GELU), trong đó lớp thứ hai có stride là 2. Sau đó, các positional embedding dạng sin được thêm vào và đưa qua các block Transformer encoder theo cấu trúc cơ bản.

Decoder sử dụng positional embedding được học và biểu diễn các token đầu vào-đầu ra. Cả encoder và decoder có cùng số lớp và chiều rộng.

Đối với tiếng Anh, sử dụng tokenizer byte-level BPE giống GPT-2. Với mô hình đa ngôn ngữ, từ vựng được điều chỉnh lại để hạn chế phân mảnh khi xử lý các ngôn ngữ không phải tiếng Anh, giúp mô hình hoạt động ổn định hơn.

**Multitask Format**

Thay vì xây dựng hệ thống ASR với nhiều thành phần riêng biệt (như nhận diện giọng nói, phân biệt người nói, chuẩn hóa văn bản,...), nhóm đề xuất một mô hình duy nhất thực hiện tất cả các tác vụ dựa trên tín hiệu âm thanh đầu vào.

Mô hình nhận vào chuỗi các token xác định nhiệm vụ: ngôn ngữ đang nói, tác vụ cần thực hiện (dịch <|translate|> hoặc phiên âm <|transcribe|>), và lựa chọn dự đoán thời gian hay không. Nếu không có giọng nói, mô hình sinh ra token đặc biệt <|nospeech|>. Khi có timestamps, mô hình dùng độ phân giải 20ms và chèn <|notimestamps|> để đánh dấu thời gian. Cuối cùng, mô hình kết thúc dự đoán bằng <|endoftranscript|>.

**Kích thước mô hình:**

Tiny: 4 layers, 384 width, 6 heads — 39M params

Base: 6 layers, 512 width, 8 heads — 74M params

Small: 12 layers, 768 width, 12 heads — 244M params

Medium: 24 layers, 1024 width, 16 heads — 769M params

Large: 32 layers, 1280 width, 20 heads — 1550M params

**Fine tuning Whisper small:**

Set up môi trường, chuẩn bị tài nguyên: Mô hình được fine tune bằng Kaggle sử dụng 2 GPU T4.

Do giới hạn về tài nguyên tính toán em sử dụng phương pháp chia nhỏ bộ data để training. Với bộ data được cung cấp bao gồm 56000 mẫu sau khi được tiền xử lý đem đi chia thành 3 phần với mỗi phần xấp xỉ 16000 mẫu.



Các bộ data nhỏ được luân phiên sử dụng để fine tune mô hình với 5 epoch mỗi bộ. Tổng cộng hết 36 giờ.

**PhoWhisper:**

PhoWhisper là một hệ thống nhận dạng tiếng nói tự động (ASR) dành riêng cho tiếng Việt, được phát triển dựa trên mô hình đa ngôn ngữ Whisper. Nhằm tăng cường độ chính xác và khả năng tổng quát của mô hình, nhóm nghiên cứu đã tinh chỉnh Whisper trên một tập dữ liệu gồm 844 giờ phát ngôn tiếng Việt với nhiều vùng phương ngữ khác nhau. PhoWhisper được cung cấp dưới 5 phiên bản khác nhau, phù hợp với nhiều nhu cầu sử dụng. Kết quả thực nghiệm cho thấy PhoWhisper đạt hiệu suất hàng đầu trên các bộ dữ liệu đánh giá ASR chuẩn cho tiếng Việt.

**Kết quả:**

**A blue and white rectangular sign with black text

AI-generated content may be incorrect.**

1. **Hậu xử lý:**
   1. **Một số lỗi sau khi sử dụng model**

Một số lỗi thường gặp:​

|  |  |  |
| --- | --- | --- |
| Audio id | ground truth | Model’s output |
| 0000144332-1 | …. ếch pơ rai rì mu … | experience |
| 0000152525-2 | … goa sinh từn… | Washington |
| 0000122189-1 | mười hai | một mươi hai |
| 0000001068-1 | Tháng tư | tháng bốn​ |
| 00001525512-2 |  | Sangtuaàpađ |

Một số lỗi đã gặp khi triển khai model:

* Các model đa ngôn ngữ như whiper hay dolphin có thể nghe được tiếng anh trong các câu tiếng việt, tuy nhiên ground truth là thuần việt
* Một số từ trong tiếng việt có thể bị hay thế bằng các từ đồng nghĩa như ( mười à một mươi , tư à bốn )
* Model có thể nối các ký tự được sinh ra thành một từ vô nghĩa.
  1. **Hướng giải quyết**

Với một số lỗi đã phát hiện như trên, nhóm đa quyết định sử dụng 2 phương pháp để thử nghiệm cải thiện kết quả:

-seq2seq:

sử dụng một model sửa lỗi chính tả cho tiếng việt: [**bmd1905**](https://huggingface.co/bmd1905)**/**[**vietnamese-correction-v2**](https://huggingface.co/bmd1905/vietnamese-correction-v2) để sửa đầu ra của model thành kết quả cuối cùng.

Ví dụ: côn viec kin doanh thì rất kho khan nên toi quyết dinh chuyển sang nghề khac ​ à Công việc kinh doanh thì rất khó khăn nên tôi quyết định chuyển sang nghề khác.

Huấn luyện SentencePiece Tokenizer sử dụng tập văn bản được trích xuất từ **transcript** của toàn bộ tập huấn luyện.

-LLMs:

Sử dụng API của gemini với prompt: để sửa đầu ra của model:

1. **Kết quả thực nghiệm**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Model​** | **Wav2vec 2.0​** | **PhoWhisper​** | **Whisper​** | **Dolphin​** |
| **work**​ | **Finetune, sử dụng decoder ngram\_lm model, LLM tách token**​ | **Infer, sử dụng LLM**​ | **Finetune, sử dụng LLM**​ | **Infer, thêm seq2seq và llm**​ |
| **Model size**​ | **base**​ | **medium**​ | **small**​ | **small**​ |
| **Pretrained/**  **Fine-tuned**​ | **Có fine-tune**​ | **Có**​ | **Có**​ | **Có/không**​ |
| **Params**​ | **95M**​ | **769M**​ | **244M**​ | **372M**​ |
| **WER – private test**​ | **49.38**​ | **37.81**​ | **49.22**​ | **50.5**​ |

* **Wav2vec 2.0:**
  + **Pretrained**
  + **Fine-tuned**​
* **Whisper**:
* **Pretrained**  
   ▪️ Whisper là mô hình nhận dạng giọng nói tự động được OpenAI huấn luyện trên hơn **680.000 giờ** dữ liệu âm thanh đa ngôn ngữ và đa tác vụ.  
   ▪️ Mô hình hỗ trợ hơn **90 ngôn ngữ**, tích hợp khả năng tự động nhận diện ngôn ngữ đầu vào, dịch sang tiếng Anh, phân đoạn lời thoại, gán thời gian và xử lý tiếng ồn cơ bản.  
   ▪️ Whisper được thiết kế theo kiến trúc **Transformer encoder-decoder**, sử dụng Mel spectrogram làm đầu vào, với tokenizer byte-level BPE.
* **Fine-tuned**  
   ▪️ Việc fine-tune mô hình Whisper được thực hiện với phiên bản **Whisper Small** bằng 2 GPU T4 trên nền tảng Kaggle.  
   ▪️ Dữ liệu bao gồm khoảng **56.000 mẫu tiếng Việt** sau khi tiền xử lý, được chia thành 3 phần nhỏ (~16.000 mẫu mỗi phần) để phù hợp với tài nguyên tính toán.  
   ▪️ Quá trình fine-tuning diễn ra qua nhiều vòng lặp, mỗi tập dữ liệu được huấn luyện trong **5 epoch**, tổng thời gian huấn luyện khoảng **36 giờ**.
* **PhoWhisper**:
* **Pretrained**  
   ▪️ PhoWhisper là mô hình nhận dạng tiếng nói dành riêng cho **tiếng Việt**, được tinh chỉnh từ Whisper đa ngôn ngữ.  
   ▪️ Mô hình được huấn luyện trên **844 giờ dữ liệu tiếng Việt** từ nhiều vùng miền, bao gồm đa dạng phương ngữ để tăng khả năng tổng quát.  
   ▪️ PhoWhisper có **5 phiên bản**, phù hợp với các nhu cầu khác nhau về độ chính xác và hiệu năng.  
   ▪️ Kết quả thực nghiệm cho thấy PhoWhisper đạt **hiệu suất hàng đầu** trên các bộ dữ liệu ASR chuẩn dành cho tiếng Việt.
* **Dolphin:**
  + **Pretrained** 
    - Tổng dữ liệu huấn luyện vượt quá **210.000 giờ audio**, bao gồm cả **dữ liệu độc quyền từ Dataocean AI** và các nguồn **mã nguồn mở khác**.
    - Dữ liệu bao phủ **40 ngôn ngữ phương Đông**, bao gồm cả **tiếng Việt**, và **22 phương ngữ tiếng Trung**, đến từ các khu vực như **Đông Á, Nam Á, Đông Nam Á** và **Trung Đông**.
    - Sự đa dạng này giúp mô hình nhận diện được **nhiều kiểu phát âm khác nhau**, từ đó **khắc phục các hạn chế** của những mô hình trước đây như **Whisper**.

**Kết luận:**

* **PhoWhisper** là mô hình cho kết quả tốt nhất với **WER = 37.81**, nhờ kích thước lớn (**medium, 769M**) và sử dụng inference kết hợp **LLM** hiệu quả.
* **Whisper** và **Wav2Vec 2.0** có kết quả WER gần nhau (**49.22** và **49.38**), dù Whisper có ít tham số hơn, cho thấy khả năng tận dụng **fine-tune + LLM** của Whisper hiệu quả.
* **Dolphin** có kích thước vừa phải (**372M**) nhưng đạt **WER cao nhất (50.5)**, mặc dù có tích hợp thêm **seq2seq + LLM**, cho thấy mô hình chưa được tối ưu hoặc không phù hợp với tập dữ liệu tiếng Việt hiện tại.
* Tổng thể, việc sử dụng **LLM** hỗ trợ inference hoặc fine-tune giúp cải thiện hiệu quả mô hình, nhưng hiệu quả còn phụ thuộc vào cấu trúc encoder, phương pháp huấn luyện, và độ phù hợp với ngôn ngữ cụ thể.