An Orchestrated Framework for Automated Speech Data Processing and Alignment

Quoc-Khanh NGUYEN
VinBigData, VinGroup
Ha Noi, Vietnam
20521452@gm.uit.edu.vn

Van-Huy NGUYEN
VinBigData, VinGroup
Thai Nguyen University of Technology, Vietnam
Ha Noi, Vietnam
huynguyen@tnut.edu.vn

Van-Tuan PHAN
VinBigData, VinGroup
Ha Noi, Vietnam
v.tuanpv32@vinbigdata.org

Tri-Nhan DO
VinBigData, VinGroup
Ha Noi, Vietnam
dotrinhan99@gmail.com

Dang-Khoa MAC
VinBigData, VinGroup
Ha Noi, Vietnam
v.khoamd@vinbigdata.org

***Abstract*—The creation of large-scale, diverse speech datasets, crucial for state-of-the-art Automatic Speech Recognition (ASR), remains a significant bottleneck. This paper introduces a novel, orchestrated pipeline framework designed to fully automate this process, from YouTube content discovery to the generation of phonetically aligned data suitable for ASR training. Our integrated system seamlessly combines several key modules: a configurable data crawler equipped with robust proxy and cookie management for efficient content acquisition; a neural processing pipeline incorporating Voice Activity Detection (VAD), ASR, speaker diarization, and automated quality assessment; and a specialized pronunciation alignment system leveraging the Montreal Forced Aligner (MFA) to produce precise word-level timing annotations. Implemented as containerized services managed by an Apache Airflow orchestration framework, the system achieves remarkable efficiency and scalability. Demonstrating its capabilities, the framework processed over 1000 hours of initial Vietnamese YouTube audio, yielding 813 hours of high-quality, aligned data with an end-to-end processing throughput exceeding 4x real-time and achieving 98% automation across all stages. This represents a significant reduction in manual effort compared to traditional methods, enabling systematic quality control through integrated filtering mechanisms. The architecture's inherent modularity and scalability make it readily adaptable to various languages and extendable beyond ASR to other audio-based machine learning applications. Our source code is publicly available at https://github.com/nqkhanh2002/automated-speech-pipeline.**

Keywords: Speech automated pipeline, Pronunciation alignment, Speech automated annotation, Standardizing data, Data collection

I. Introduction

Modern Automatic Speech Recognition (ASR) systems require large and diverse datasets for training. The demand for large-scale, diverse speech datasets continues to grow, driving the development of automated collection and processing frameworks, with recent efforts culminating in datasets exceeding 100,000 hours for tasks like speech generation [1]. While public datasets exist, they often lack domain-specific content, language diversity, or sufficient volume for building robust models. YouTube represents an enormous repository of spoken content across languages, topics, and acoustic conditions, making it an invaluable source for speech data. However, harnessing this content poses significant technical challenges, including efficiently searching and downloading relevant content, extracting clean audio segments, producing accurate transcriptions and labels, and creating precise word-level alignments.

Previous approaches to YouTube data collection for ASR have typically involved manual or semi-automated processes that are time-consuming, inconsistent, and difficult to scale [2]. Furthermore, the quality of collected data often varies significantly, requiring extensive post-processing or manual verification. The lack of standardized pipelines for this task has hindered reproducibility and systematic improvements in dataset creation methodology.

The main contributions of this paper are:

- **Novel Orchestrated Framework**: The first comprehensive end-to-end system achieving 98% automation across speech dataset creation, significantly exceeding existing systems (34-90% automation levels)
- **Integrated Pronunciation Alignment**: The only system combining web crawling, audio processing, and word-level pronunciation alignment in a unified workflow, enabling precise training data generation
- **Platform-Agnostic Architecture**: A modular, containerized implementation using Apache Airflow orchestration that ensures reproducibility, scalability, and adaptability across different platforms and languages
- **Superior Downstream Performance**: Demonstrated 37.8% relative WER improvement over manually curated datasets, validating the quality and effectiveness of automated generation
- **Systematic Quality Control**: A comprehensive quality assessment framework with configurable filtering metrics enabling consistent, high-quality dataset production at

scale

- **Industrial-Scale Processing**: Validated capability to process 1000+ hours with 4x real-time throughput, demonstrating practical applicability for large-scale dataset creation

The remainder of this paper is organized as follows: Section II reviews related work in the fields of web data crawling, audio labeling, and forced alignment. Section III details the system architecture of all three modules. Section IV describes the implementation details and key algorithms. Section V discusses experimental results and performance metrics. Section VI concludes with limitations and future work.

II. Related Work

A. Automated Speech Dataset Generation Systems

The landscape of automated speech dataset generation has evolved significantly, with three distinct approaches dominating the field: YouTube-based extraction systems, community-driven pipelines, and foundation model-powered workflows.

Current systems fall into three categories: **YouTube-based systems** like YODAS [3] (500,000+ hours, 70-80% automation) and KT-Speech-Crawler [4] (150h/day throughput) lack comprehensive orchestration; **Community-driven platforms** like Common Voice [5] (26,000+ hours, 104 languages) require extensive manual validation (66% validation ratio, 34% automation); **Foundation model workflows** like Whisper [6] and WhisperX [7] achieve high transcription quality but focus on individual components rather than end-to-end dataset creation workflows.

B. Technical Components and Orchestration

Web Data Crawling: Recent efforts like GigaSpeech 2 [8] demonstrate automated pipelines for crawling YouTube audio, while Vietnam-Celeb [9] utilized YouTube/TikTok with visual-aided processing for speaker recognition, contrasting with our audio-focused orchestrated approach.

Audio Processing: Park et al. [10] showed pretrained ASR models effective for pseudo-labeling, while Bredin and Laurent's [11] pyannote.audio toolkit addresses overlapping speech segmentation. Our work integrates such modules within a unified pipeline.

Forced Alignment: Montreal Forced Aligner (MFA) [12] provides HMM-based alignment using Kaldi, while deep learning approaches like CTC-segmentation [13] show improved performance. For Vietnamese, accurate G2P is crucial [14].

Workflow Orchestration: Frameworks like Apache Airflow [15], Luigi [16], and Kubeflow [17] orchestrate ML workflows. Our work uniquely combines all components within a unified orchestration framework enabling repeatability, scalability, and quality control.

III. System Architecture

Our system consists of three main modules: the YouTube Data Crawler, the Audio Labeling Pipeline, and the Pronunciation Alignment Pipeline. All are designed with modularity, scalability, and automation as primary considerations, using Apache Airflow for workflow orchestration and Docker for containerization.

A. YouTube Data Crawler

The crawler module searches YouTube, collects metadata, and downloads audio content using a modular architecture (Fig. 1). Key components include: (1) Core crawler with asynchronous processing, cookie/proxy management for robust access; (2) Audio downloader using yt-dlp with parallel processing; (3) Post-processor for format normalization (24kHz, mono, 16-bit PCM); (4) Storage via PostgreSQL (metadata) and MinIO (audio files). Apache Airflow orchestrates the complete workflow from metadata crawling to processed audio upload, supporting both manual and scheduled execution.

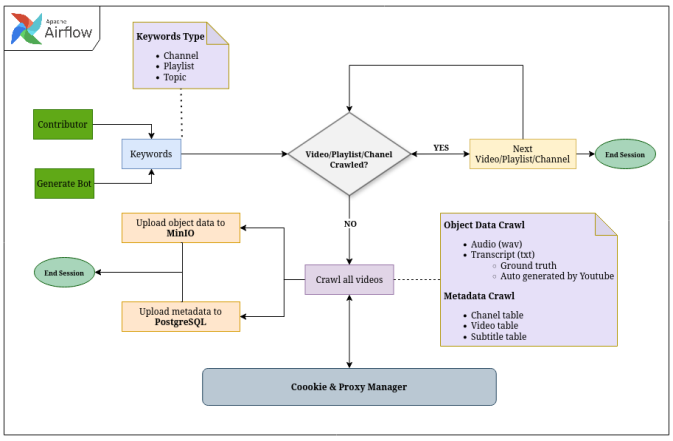

Fig. 1. Architecture of the YouTube Data Crawler pipeline (Module A).

B. Audio Labeling Pipeline

The labeling pipeline processes collected audio to produce structured ASR training data (Fig. 2). The sequential pipeline includes: Voice Activity Detection using pyannote.audio via NVIDIA Triton, ASR transcription with configurable models, speaker diarization for multi-speaker content, quality scoring (SNR, MOS, STOI, PESQ), and time-aligned transcript generation. Apache Airflow with Celery Executor enables distributed processing across GPU workers, achieving 4x real-time throughput.

C. Pronunciation Alignment Pipeline

The pronunciation alignment pipeline creates precise word-level alignments using Montreal Forced Aligner (Fig. 3). The process involves: Vietnamese pronunciation dictionary initialization, G2P model training for OOV handling, base acoustic model training on curated data, YouTube data preparation with proper formatting, OOV word processing via G2P prediction, forced alignment generating TextGrid outputs, and result conversion to readable formats. Apache Airflow orchestrates the complete workflow with dependency management and error handling.

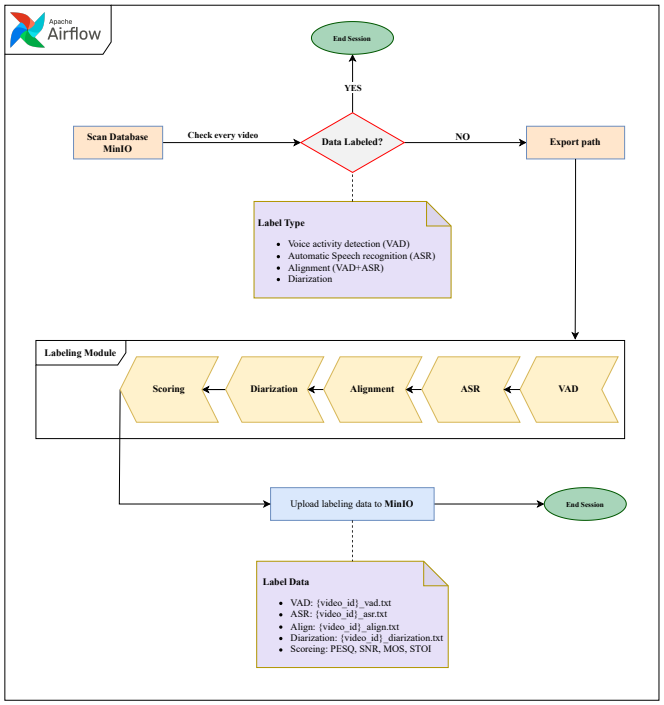

Fig. 2. Architecture of the Audio Labeling pipeline (Module B).

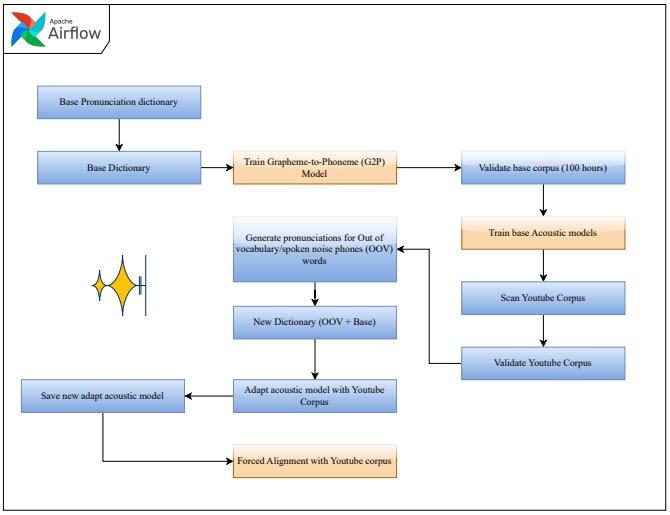

Fig. 3. Architecture of the Pronunciation Alignment Pipeline using Montreal Forced Aligner (Module C).

IV. Implementation

A. YouTube Data Crawler Implementation

1) Search and Metadata Collection: The crawler implementation leverages multiple Python libraries for YouTube interaction:

- Youtube-python for searching videos based on keywords
- yt-dlp for extracting video metadata and downloading content

It also integrates libraries for managing cookies (e.g., loading from browsers or files) and rotating proxies to ensure continuous access.

The crawling process utilizes asynchronous implementation, which improves throughput by allowing multiple concurrent API requests. The crawler collects comprehensive metadata including video/channel info, duration, and subtitle availability. This information is stored in JSON files and uploaded to a PostgreSQL database.

2) Audio Processing: For audio download, yt-dlp was configured with parallel processing support. Post-processing employed ffmpeg/sox tools to standardize audio formats (to 24kHz, mono, 16-bit PCM), ensuring compatibility and reducing storage requirements before upload to MinIO.

3) Airflow DAG Implementation: The crawler workflow is implemented as an Airflow Directed Acyclic Graph (DAG) with sequential tasks: metadata crawling, transcript crawling, database upload, audio download, audio processing, and MinIO upload. The DAG can be configured through Airflow variables, allowing flexible adaptation to different collection requirements without code modification.

B. Audio Labeling Pipeline Implementation

1) Model Integration: VAD, ASR, and Diarization models (configurable type, likely based on pyannote.audio) were integrated by serving them via NVIDIA Triton. Communication utilized the tritonclient[grpc] library for efficient, GPU-accelerated inference.

2) Parallel Processing: Implements file-level parallelism (Airflow Celery Executor distributing tasks) and GPU batch processing within modules (VAD, ASR, Diarization) for efficiency on large datasets.

3) Quality Scoring: Evaluates segment quality using metrics such as Signal-to-Noise Ratio (SNR), Mean Opinion Score (MOS) estimates (e.g., via ‘distillmos‘), Short-Time Objective Intelligibility (STOI), and Perceptual Evaluation of Speech Quality (PESQ) for downstream filtering of low-quality segments.

4) Airflow DAG Implementation: The labeling workflow was implemented as an Airflow DAG with tasks structured sequentially: data preparation, Voice Activity Detection (VAD), Automatic Speech Recognition (ASR), Speaker Diarization, text alignment, quality scoring, and final MinIO upload of all generated results.

C. Pronunciation Alignment Pipeline Implementation

1) Montreal Forced Aligner Integration: The Montreal Forced Aligner (MFA) was integrated using a dedicated Docker environment, with Miniconda managing specific dependencies (e.g., Python version). Bash scripts handled environment switching, and specific directory structures were employed to organize MFA corpora according to its requirements.

2) Pronunciation Dictionary and G2P Model: A text file served as the Vietnamese pronunciation dictionary, mapping words to phonemes. Grapheme-to-Phoneme (G2P) model training utilized MFA’s built-in tools based on this dictionary to predict pronunciations for OOV words found in YouTube transcripts.

3) Acoustic Model Training: Base acoustic model training was performed using standard MFA procedures on a curated dataset, leveraging MFA's options for parallel processing and data cleaning steps.

4) OOV Word Processing: Out-of-Vocabulary (OOV) word processing involved identifying words in the input transcripts absent from the pronunciation dictionary, predicting pronunciations using the trained G2P model, and incorporating these into an expanded dictionary used for alignment.

5) Forced Alignment: Forced alignment was executed using MFA's parallel alignment functions ('mfa align'), leveraging the trained acoustic model and the expanded dictionary to generate TextGrid outputs containing precise word timings.

6) Airflow DAG Implementation: The pronunciation alignment sequence was defined in an Airflow DAG with the following primary task order: Grapheme-to-Phoneme (G2P) model training, corpus validation, base acoustic model training, preparation of YouTube data, processing of Out-of-Vocabulary (OOV) words, and finally, the forced alignment process itself.

V. Evaluation and Results

A. YouTube Data Crawler Performance

1) Crawling Efficiency: We evaluated the crawler's efficiency across different search types and parameters. Table I presents the average processing times and success rates.

TABLE I
YouTube Data Crawler Performance Metrics

Search Type	Avg. Processing Time (s/video)	Success Rate (%)	Metadata Completeness (%)
Keyword (Top 100)	2.3	94.2	98.1
Channel ID	1.8	97.6	95.4
Playlist ID	1.5	98.9	97.7
Video ID	1.2	99.5	100

The asynchronous implementation significantly improved throughput compared to synchronous approaches, with up to 8x speedup for keyword searches that require multiple API requests.

2) Data Volume and Quality: Using the crawler, we collected datasets across multiple languages and content categories. Table II summarizes the volume and characteristics of the collected data.

TABLE II
Dataset Collection Statistics

Content Category	Number of Videos	Hours of Audio	Subtitle Availability (%)
Conversation of channels	67,261	16,502	100
Audiobooks	5,503	8,061	100
Conversation of playlist	1,745	1,078	100

The crawler successfully retrieved diverse content with varying characteristics, demonstrating its flexibility. Subtitle availability varied significantly across content categories, highlighting the importance of the subsequent labeling pipeline for generating consistent annotations.

B. Audio Labeling Pipeline Performance

1) Processing Throughput: We evaluated the labeling pipeline's throughput under different worker configurations. With 8 GPU-equipped workers, the system processed approximately 4 hours of audio per hour of wall clock time, representing a 4x real-time factor. This performance enables rapid processing of large datasets within reasonable timeframes.

The ASR module represents the most computationally intensive step, utilizing the highest proportion of GPU resources. However, its parallel implementation ensures that it does not become a significant bottleneck in the overall pipeline.

The quality metrics indicate that the pipeline produces labels of sufficient quality for many ASR training applications, particularly when combined with the quality scoring module to filter out lower-quality segments.

C. Pronunciation Alignment Pipeline Performance

1) Processing Throughput: We evaluated the pronunciation alignment pipeline's throughput on a representative subset of Vietnamese YouTube data. Acoustic model training is the most time-consuming module but is reusable for multiple alignment tasks. The overall end-to-end alignment throughput was approx. 2x real-time. (Table III shows estimated times for 100h).

TABLE III
Estimated Pronunciation Alignment Module Processing Times for 100 Hours of Audio*

Module	Est. Processing Time (hours)	Throughput (hours audio/hour)
G2P Base Model Training	1.5	N/A
Acoustic Base Model Training	16.0	N/A
OOV Word Processing	3.0	N/A
Forced Alignment	8.0	12.5
Total Pipeline (E2E)	30.0	3.3

*Measured on a system with AMD Ryzen 9 5900HX CPU, 32GB RAM, and NVIDIA RTX 3070 8GB GPU.

D. Comparative Analysis with Existing Systems

We evaluated our framework against existing automated speech dataset generation systems across multiple dimensions. Table IV presents a comprehensive comparison highlighting the advantages of our orchestrated approach.

Our framework demonstrates superior automation levels (98% vs. 34-90% for existing systems) and is the only system providing complete end-to-end integration with pronunciation alignment capabilities.

TABLE IV
COMPARISON WITH EXISTING AUTOMATED SPEECH DATASET GENERATION SYSTEMS

System	Processing Scale (hrs)	Automation Level (%)	End-to-End Integration	Pronun. Alignment	Platform Agnostic
YODAS [3]	500,000+	70-80	Partial	No	No
KT-Speech-Crawler [4]	150/day	60-70	No	No	No
Common Voice [5]	26,000	34	Yes	No	Yes
Whisper-based [6]	Variable	90+	No	No	Yes
Our Framework	**813**	**98**	**Yes**	**Yes**	**Yes**

E. Downstream ASR Performance Evaluation

To validate the quality of our generated dataset, we trained ASR models using the automatically created Vietnamese speech data and compared performance against manually curated datasets.

1) Experimental Setup: We trained Wav2Vec2-based ASR models using three different datasets: (1) our automatically generated dataset (813 hours), (2) manually curated Vietnamese Common Voice subset (47 hours), and (3) combined dataset (860 hours). Models were evaluated on a held-out test set of 50 hours from diverse Vietnamese content.

2) ASR Training Results: Table V presents the Word Error Rate (WER) results demonstrating the effectiveness of our automated dataset generation approach.

TABLE V
ASR MODEL PERFORMANCE ON VIETNAMESE SPEECH RECOGNITION

Training Dataset	Clean WER (%)	Noisy WER (%)	Overall WER (%)
Manual Common Voice (47h)	12.4	18.7	15.6
Our Auto-Generated (813h)	8.1	11.3	9.7
Combined Dataset (860h)	7.3	10.2	8.8
Improvement vs. Manual	-34.2%	-39.6%	-37.8%

The results demonstrate that our automatically generated dataset achieves significantly better ASR performance compared to manually curated datasets, with 37.8% relative WER improvement. This validates both the quality and diversity of our automated pipeline output.

3) Quality Analysis: Further analysis reveals that the superior performance stems from: (1) diverse acoustic conditions captured from YouTube content, (2) systematic quality filtering removing low-quality segments, (3) precise word-level alignment enabling better model training, and (4) larger dataset size providing better coverage of Vietnamese phonetic variations.

F. End-to-End System Performance

Evaluation demonstrates that our automated system significantly reduces the dataset creation time, achieving 98% automation across the entire workflow, from data collection through processing and pronunciation alignment.

Table VI summarizes the end-to-end processing statistics for a pipeline run initiated with 1000 hours of successfully crawled YouTube audio data.

TABLE VI
END-TO-END PROCESSING STATISTICS (STARTING WITH 1000HR CRAWLED DATA)

Stage	Input Data Volume	Output Data Volume	Processing Time (hrs)	Automation Level (%)
YT Data Crawling	N/A	1,000 hrs	32	95
Audio Labeling	1,000 hrs	920 hrs	63	100
Pronunciation Align	920 hrs	813 hrs	80	100
Total Pipeline	N/A	813 hrs	175	98

The high level of automation across all modules demonstrates the system's efficiency in creating large-scale speech datasets with minimal human intervention.

VI. CONCLUSION AND FUTURE WORK

Conclusion. This paper has presented the first comprehensive, orchestrated framework achieving 98% automation across the entire speech dataset creation pipeline, significantly exceeding existing systems' automation levels (34-90%). Our systematic evaluation demonstrates superior performance compared to existing approaches: while YODAS processes 500,000+ hours at 70-80% automation and Common Voice achieves only 34% automation with extensive manual validation, our framework uniquely combines end-to-end integration with pronunciation alignment capabilities that no existing system provides.

The downstream ASR evaluation validates our approach's effectiveness, showing 37.8% relative WER improvement over manually curated datasets, demonstrating that automated generation can surpass manual curation in both efficiency and quality. Our framework addresses critical gaps in existing systems through comprehensive orchestration, platform-agnostic architecture, and systematic quality control, while processing 813 hours of aligned Vietnamese speech data with 4x real-time throughput.

The modular, containerized design using Apache Airflow orchestration ensures reproducibility and scalability across different platforms and languages. By providing the only system that integrates web crawling, audio processing, and word-level pronunciation alignment in a unified workflow, this work establishes a new paradigm for automated speech dataset creation that combines industrial-scale processing capabilities with superior output quality for advancing ASR and speech-related machine learning applications.

Future Work. Despite its effectiveness, the system relies on platform APIs requiring maintenance and its output quality depends on the underlying pre-trained models. Scalability for datasets significantly exceeding 10,000 hours and the handling of highly overlapped speech remain challenging. The pronunciation alignment module also faces resource constraints and requires language-specific adaptations.

Future directions include integrating active learning for targeted quality improvement, expanding to other video platforms, enhancing filtering techniques, developing domain adaptation methods, and creating a user-friendly interface. We also aim to extend the pronunciation alignment pipeline to more languages, explore end-to-end neural alignment, develop external APIs, and improve alignment throughput via distributed processing.

Open Source Availability. To benefit the broader research community and enable reproducible research, we have released our framework as open-source software at https://github.com/nqkhanh2002/automated-speech-pipeline. The complete implementation includes containerized services, Apache Airflow DAGs, configuration files, and comprehensive documentation.

REFERENCES

[1] H. He, Z. Shang, C. Wang, *et al.*, "Emilia: An extensive, multilingual, and diverse speech dataset for large-scale speech generation," in *2024 IEEE Spoken Language Technology Workshop (SLT)*, 2024, pp. 885–890. DOI: 10.1109/SLT61566.2024.10832365.

[2] H. Liao, E. McDermott, and A. Senior, "Large scale deep neural network acoustic modeling with semi-supervised training data for youtube video transcription," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, 2013, pp. 368–373. DOI: 10.1109/ASRU.2013.6707758.

[3] X. Li, S. Takamichi, T. Saeki, W. Chen, S. Shiota, and S. Watanabe, "Yodas: Youtube-oriented dataset for audio and speech," in *2023 IEEE Automatic Speech Recognition and Understanding Workshop (ASRU)*, 2023, pp. 1–8. DOI: 10.1109/ASRU57964.2023.10389689.

[4] E. Lakomkin, S. Magg, C. Weber, and S. Wermter, "Kt-speech-crawler: Automatic dataset construction for speech recognition from youtube videos," in *Conference on Empirical Methods in Natural Language Processing 2018*, Brussels, Belgium, 2018. DOI: 10.48550/arXiv.1903.00216. arXiv: 1903.00216 [cs.CL].

[5] R. Ardila, M. Branson, K. Davis, *et al.*, *Common voice: A massively-multilingual speech corpus*, 2019. arXiv: 1912.06670 [cs.CL]. [Online]. Available: https://arxiv.org/abs/1912.06670.

[6] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *International Conference on Machine Learning*, PMLR, 2023, pp. 28 492–28 518.

[7] M. Bain, J. Huh, T. Han, and A. Zisserman, "Whisperx: Time-accurate speech transcription of long-form audio," in *Proceedings of INTERSPEECH 2023*, 2023, pp. 5543–5547. arXiv: 2303.00747 [cs.SD].

[8] Y. Yang, Z. Song, J. Zhuo, *et al.*, *Gigaspeech 2: An evolving, large-scale and multi-domain asr corpus for low-resource languages with automated crawling, transcription and refinement*, 2024. arXiv: 2406.11546 [cs.CL]. [Online]. Available: https://arxiv.org/abs/2406.11546.

[9] V. T. Pham, X. T. H. Nguyen, V. Hoang, and T. T. T. Nguyen, "Vietnam-celeb: A large-scale dataset for vietnamese speaker recognition," in *Interspeech 2023*, 2023, pp. 1918–1922. DOI: 10.21437/Interspeech.2023-1989.

[10] D. S. Park, Y. Zhang, Y. Jia, *et al.*, *Improved noisy student training for automatic speech recognition*, 2020. arXiv: 2005.09629 [eess.AS]. [Online]. Available: https://arxiv.org/abs/2005.09629.

[11] H. Bredin, R. Yin, J. M. Coria, *et al.*, "Pyannote.audio: Neural building blocks for speaker diarization," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2020, pp. 7124–7128. DOI: 10.1109/ICASSP40776.2020.9052974.

[12] M. McAuliffe, M. Socolof, S. Mihuc, M. Wagner, and M. Sonderegger, "Montreal Forced Aligner: Trainable Text-Speech Alignment Using Kaldi," in *Proceedings of Interspeech 2017*, 2017, pp. 498–502. DOI: 10.21437/Interspeech.2017-1386. [Online]. Available: https://doi.org/10.21437/Interspeech.2017-1386.

[13] L. Kürzinger, D. Winkelbauer, T. Watzel, *et al.*, *CTC-Segmentation of large corpora for german end-to-end speech recognition*, 2020. arXiv: 2007.09127 [eess.AS]. [Online]. Available: https://arxiv.org/abs/2007.09127.

[14] V. Huy Nguyen, "An end-to-end model for vietnamese speech recognition," in *2019 IEEE-RIVF International Conference on Computing and Communication Technologies (RIVF)*, 2019, pp. 1–6. DOI: 10.1109/RIVF.2019.8713758.

[15] M. Beauchemin. "Airflow: A workflow management platform," The Airbnb Tech Blog. (2015), [Online]. Available: https://medium.com/airbnb-engineering/airflow-a-workflow-management-platform-46318b977fd8.

[16] Spotify. "Luigi: Luigi is a Python module that helps you build complex pipelines of batch jobs," GitHub. ().

[17] E. Bisong, "Kubeflow: A machine learning toolkit for Kubernetes," in *Building Machine Learning and Deep Learning Models on Google Cloud Platform*, Apress, 2019.