Fine-Tuning OpenAI Whisper-Small for Domain-Specific Medical Speech Recognition within a Microservice Architecture

Alaeddine Moussa[1], Noursene Drine[2]
[1]Université de la Manouba, École Nationale des Sciences de l'Informatique, Tunisia
[2]Universite de Tunis El Manar, École Nationale des Sciences de l'Informatique, Tunisia
E-mail: dr.alaeddine.moussa@gmail.com, noursene.drine@corilus.be

Keywords: Speech recognition, Whisper, fine-tuning, healthcare, microservices

Received:

We fine-tune Whisper-small (244M parameters) on 8.5 hours of in-domain medical audio and evaluate with word error rate (WER). Compared to an unadapted Whisper-small baseline, our fine-tuned model reduces WER from ∼63% to ∼32%. While the relative gain is substantial, this accuracy is not suitable for unsupervised clinical use; we position the system as a clinician-in-the-loop assistant. We also describe deployment as an on-premise microservice and report latency/throughput considerations.

Povzetek: Študija pokaže, da fino-uglaševanje Whisper-small na omejenem medicinskem zvoku občutno zniža WER, vendar ostane primerno predvsem za virtualnega pomočnika z lokalno mikroservisno uvedbo.

1 Introduction

Healthcare professionals still face difficulties in organizing massive patient data, managing administrative tasks, and making sure that bills are handled accurately. Inefficiencies, lost time and errors can result from manual entries and the wrong management of data and patient records which can eventually affect communication and standard care. Furthermore, it can be challenging and time-consuming to assess financial performance and spot trends without the right tools for data visualization[1]. However, deploying speech recognition in clinical practice is challenging due to technical accuracy requirements and domain-specific vocabulary. Previous studies of automatic speech recognition (ASR) in clinical contexts have reported widely varying accuracy (WER ranging from 18% up to 63% in different systems and environments[1]). This variance stems from differences in audio quality, medical jargon, and speaker accents, underlying the need for tailored STT solutions in healthcare. Recent advances in deep learning have led to powerful general-purpose speech models; for example, OpenAI Whisper was trained on 680,000 hours of multilingual audio (65% English, the rest spanning 98 languages)[2]. Whisper's transformer-based architecture has demonstrated robust performance across languages, accents, and noisy inputs [3]. These characteristics make it a promising candidate for medical transcription, where audio conditions and dialects can vary.

This study presents a software platform for medical practice management, which includes multiple microservices (user management, patient records, data visualization, etc.). Within this platform, we developed a dedicated Speech-to-Text microservice to handle transcription of recorded consultations. Figure 1 illustrates the overall architecture. The STT microservice operates alongside others behind a common API Gateway, enabling independent development and scaling of the transcription component.

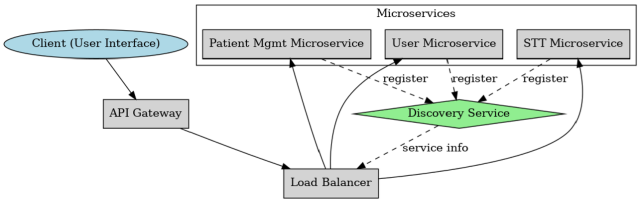


Figure 1: Global architecture of the medical software system, highlighting the Speech-to-Text (STT) microservice and its interactions. The client application sends audio recordings via an API Gateway, which routes requests through a Load Balancer to the STT microservice. The STT service registers with a Discovery Service for service lookup, and works in parallel with other microservices (User management, Patient management) within the platform.

In this paper, we focus on the design and implementation of the STT microservice using the Whisper model for medical speech recognition. In the next section, we review related STT solutions in healthcare, including commercial systems and recent research, to position our approach. We then detail our system methodology, wrapping the microservice design, model integration, and fine-tuning procedure. The dataset used for training (a public medical speech corpus from Kaggle) is described, along with the preprocessing steps undertaken to improve audio quality. We present experimental results evaluating the fine-tuned model's performance (using WER as the metric) and discuss how it compares to baseline models and human-level

performance. Finally, we address challenges encountered, such as limited training data and domain-specific errors, and suggest future improvements (e.g., larger models, more data, and error correction mechanisms) before concluding the study.

Research questions: We structure our study around four questions:

1. **RQ1:** What is the baseline WER of an unadapted Whisper-small model on our in-domain medical audio?
2. **RQ2:** To what extent does full fine-tuning reduce WER on this dataset?
3. **RQ3:** Under which conditions is the system suitable for clinician-in-the-loop use, and where does it fail?
4. **RQ4:** How can a microservice deployment meet practical constraints (latency, throughput, scaling)?

Contributions:

- A fine-tuned Whisper-based clinical ASR microservice with a WER reduction from 63% to 32%.
- A clear positioning of the system as *human-in-the-loop* for safe clinical workflows.
- A practical discussion of deployment trade-offs (interface, scaling) and a roadmap to robustness and generalization.
- Reproducible implementation details to ease adoption in healthcare settings.

2 Literature review

Speech recognition technology for healthcare applications has advanced significantly, evolving from early dictation systems to modern end-to-end deep learning models. Historically, products like Nuance Dragon Medical [4] have dominated clinical dictation by offering specialized vocabularies for physicians and demonstrating high accuracy. A recent pilot study found that a fine-tuned Whisper model performed comparably to Dragon for dictating neurosurgery reports. The Whisper model achieved a mean Word Error Rate (WER) of 1.75%, while Dragon recorded a WER of 1.54%, indicating that Whisper's performance is statistically non-inferior. Furthermore, Whisper outperformed Dragon in some linguistic aspects, with a WER of 0.50% compared to Dragon's 1.34% when excluding formatting differences. This suggests that, with appropriate tuning, open models can achieve accuracy levels similar to established proprietary systems in controlled environments. Additionally, cloud-based speech-to-text (STT) services from companies like Google, Amazon, and Microsoft have also been utilized for medical transcription. For instance, Google's Speech-to-Text API offers a dedicated healthcare model, with studies reporting WERs of approximately 8–12% for typical clinical recordings under ideal conditions. However, accuracy may decline when faced with accents or noisy environments. Concerns about data privacy and cost are also driving a preference for on-premise or open-source alternatives.

Parallels with adaptive control: Beyond speech recognition, adaptation to uncertainty is a central theme in complex dynamical systems. In adaptive and robust control, models are updated (or controllers are synthesized) to maintain performance under variability and noise (e.g., adaptive fuzzy schemes, output-feedback and neural adaptive control). *Our fine-tuning of Whisper plays an analogous role*: the base ASR is specialized to the medical domain so that decision boundaries reflect domain-specific phonetics and terminology. While these control-theoretic methods and ASR are not directly comparable in metrics or problem setting, they share the core idea that *targeted adaptation* to operating conditions yields large performance gains. See, for instance, representative works on adaptive fuzzy control for fixed-time synchronization, projective lag-synchronization with uncertainties, robust neural adaptive control for uncertain nonlinear multivariable systems, adaptive backstepping for SISO nonlinear plants and flexible manipulators, and nonlinear optimal control in industrial settings [5, 6, 7, 8, 9, 10].

Open-source and research models

Academic and open-source automatic speech recognition (ASR) frameworks such as Kaldi[11], Mozilla DeepSpeech[12], and Wav2Vec 2.0[13] have been utilized to develop medical transcription models. These models often require a significant amount of domain-specific training data. The release of OpenAI's Whisper in late 2022 marked a significant advancement in this area by introducing a pre-trained, general-purpose model equipped with strong zero-shot transcription capabilities. Whisper was trained in a fully supervised manner on a vast and diverse dataset, which gives it robustness against a variety of speech patterns and background noises [14, 15] . Whisper utilizes a Transformer encoder-decoder architecture Figure2, which allows it to not only transcribe speech but also perform speech translation and language identification using special tokens. Notably, Whisper processes audio in 30-second segments, converting input waveforms into log-Mel spectrogram features that the encoder processes, while the decoder generates text tokens autoregressively. This single-model, multitask design simplifies the speech processing pipeline [16].

Whisper's accuracy for medical dialogue can be moderate without fine-tuning. For example, the Word Error Rate (WER) of Whisper's small model is about 63% when evaluated on a general English speech corpus like Common Voice, which is too high for reliable clinical transcription.

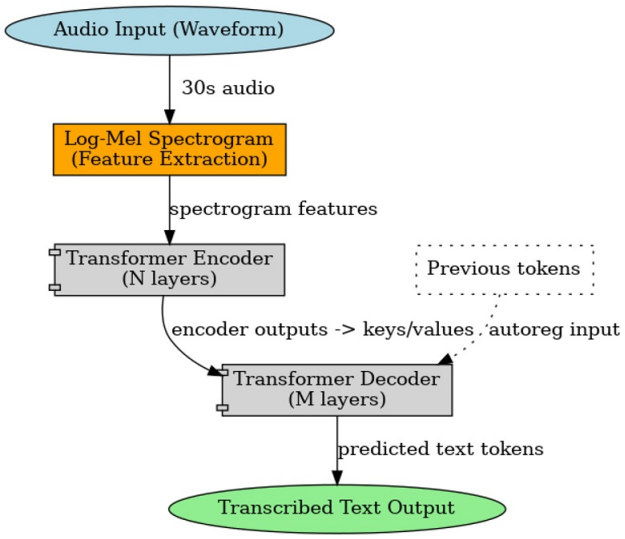


Figure 2: Whisper model architecture (Transformer-based encoder-decoder). The 30 s audio input is first converted into a log-Mel spectrogram, which the Transformer encoder ingests. The Transformer decoder then generates the transcribed text, autoregressively predicting tokens while attending to the encoder outputs. Whisper uses special tokens to handle tasks like language detection and translation as well. This architecture enables a single model to perform robust speech recognition across many languages and conditions.

To improve the recognition of medical terminology and diverse speaker styles, fine-tuning the model on in-domain data is essential. Recent research has focused on adapting these models to specific domains. One approach involves generating synthetic medical speech data for training. For instance, [15] used real-world home healthcare recordings to evaluate various Automatic Speech Recognition (ASR) systems, highlighting performance disparities among different patient groups. In another study, large-scale data augmentation created a synthetic medical speech dataset of approximately 5,486 hours. Models fine-tuned on this dataset, such as a Conformer RNN-T and a Whisper-derived model, achieved WERs as low as 3.9% to 4.6% on medical test sets. These outcomes illustrate that with sufficient domain-specific training, state-of-the-art models can reach error rates of just a few percent.

Moreover, specialized ASR providers have developed models tailored for healthcare. For example, Deepgram's Nova model, which focuses on medical training, has been reported to slightly outperform Whisper on healthcare transcription benchmarks [17]. Despite this, Whisper's openness and adaptability make it an appealing option for institutions that prefer an on-site solution without sharing data with third-party APIs[18].

To ground our work in the literature, Table 1 contrasts Dragon, Google STT, Whisper variants, and Wav2Vec 2.0 across dataset scale, WER, domain, and deployment setting.

In summary, the literature suggests that: (1) achieving high accuracy in medical speech recognition is possible, but typically requires either proprietary solutions or extensive domain training; (2) Whisper serves as a robust foundation that can be fine-tuned for medical applications, providing a balance between open-source flexibility and commercial-level accuracy; and (3) a microservice architecture can facilitate the integration of such a speech-to-text engine into healthcare software, allowing for modular deployment and scaling. Building on these insights, our work fine-tunes the Whisper model using a moderately sized public medical speech dataset and incorporates it as a microservice in a telehealth application. The following sections will detail our methodology and how it addresses the identified challenges.

3 Methodology

3.1 System architecture and integration

The speech-to-text (STT) functionality is implemented as an independent microservice within our healthcare application. We chose a microservice architecture for its modularity and scalability; each service encapsulates specific functionality and can be developed, deployed, and scaled independently. In this case, the STT microservice coexists with user management and patient record microservices, among others.

All microservices register with a central Discovery Service, allowing them to be located by name. A reverse proxy API Gateway routes external requests to the appropriate service instance. A load-balancing mechanism ensures that if multiple instances of the STT service are running (to process many audio streams simultaneously), requests are evenly distributed. Authentication is managed upstream by the gateway, allowing the STT service to focus on its core task.

Within the STT microservice, we have implemented a RESTful API to handle transcription requests. Clients, such as the web or mobile front-end used by doctors, submit audio recordings (for example, of patient consultations) via an HTTP request to the API Gateway, which forwards the requests to the STT service's endpoint. The STT service processes the audio and returns the text transcription. Additionally, the service provides endpoints for CRUD (Create, Read, Update, Delete) operations on the transcript data, enabling users to retrieve, update, or delete saved transcriptions.

This design fits a microservice architecture composed of independently deployable services. For each feature: a User service for authentication, a Patient service for patient data, and an STT service for transcription. Decoupling the STT component ensures that intensive audio processing tasks do not interfere with the responsiveness of other parts of the application. It also allows the STT model to be hosted on hardware suited for the task (such as a GPU), separate from the main application server.

Table 1: Comparative summary (figures are not strictly comparable across rows; datasets and conditions differ).

Model	Dataset size / type	Reported WER	Domain	Deployment
Dragon Medical One (Nuance)	Dictation (vendor cases)	n/a (vendor "∼99% accuracy", not WER)	Medical dictation	Cloud, proprietary
Google STT — Medical Conversation	36 primary-care conversations (re-enacted studio)	∼9.1% (medical conv.); 8.8% (general)	Medical conversation	Cloud API
Whisper-large (zero-shot)	LibriSpeech test-clean (general audiobooks)	2.5% (zero-shot, clean)	General	On-prem/open-source
Wav2Vec 2.0 Large (supervised)	LibriSpeech 960 h (general audiobooks)	1.8% / 3.3% (clean/other)	General	On-prem/open-source
Whisper-small (baseline, ours)	8.5 h in-domain (medical)	63%	Medical (ours)	On-prem/open-source
Whisper-small (fine-tuned, ours)	8.5 h in-domain (medical)	32%	Medical (ours)	On-prem/open-source

Communication between microservices utilizes HTTP requests facilitated by the gateway, and all services share a common identity/authentication scheme using JWT tokens. The STT service was developed in Python using the Flask framework for the API, as the Whisper model and its training code are also in Python, leveraging PyTorch and the Hugging Face Transformers libraries for model management. Other services were implemented in C# (.NET Core) according to the project's technology choices. However, this heterogeneity is well-supported by the microservice architecture. As long as each service exposes REST endpoints and registers with the discovery service, the specific implementation language is irrelevant to the client. This flexibility exemplifies one of the key benefits of microservices.

3.2 Whisper model integration

At the core of the STT microservice is the OpenAI Whisper model. We have integrated the small version of Whisper, which strikes a good balance between accuracy and efficiency for our use case. This small model, composed of 244M parameters, requires approximately 2 GB of VRAM for inference, fitting well within our available GPU memory. We selected this model size due to our limited computational resources and the relatively small dataset we had for fine-tuning.

While larger Whisper models, with up to 1.5B parameters, may provide higher accuracy, they also demand significantly more memory and computational power. This makes them more challenging to deploy in our real-time service without specialized hardware.

Table 2 outlines the number of parameters in each Whisper model along with their computational requirements.

Table 2: Whisper models and their approximate memory requirements

Size	Parameters	English-only	Multilingual	VRAM	Speed
tiny	39M	`tiny.en`	`tiny`	∼1 GB	∼32x
base	74M	`base.en`	`base`	∼1 GB	∼16x
small	244M	`small.en`	`small`	∼2 GB	∼6x
medium	769M	`medium.en`	`medium`	∼5 GB	∼2x
large	1550M	N/A	`large`	∼10 GB	1x

We began with the pre-trained multilingual Whisper-small checkpoint as our foundation. Whisper's architecture is based on a Transformer encoder-decoder model. The encoder transforms audio features into latent representations, while the decoder generates text. This model was initially trained on a vast corpus of voice data with transcripts that included various languages and tasks. This pre-training endows the model with strong general speech recognition capabilities, although it is not specifically optimized for medical terminology. Our goal was to fine-tune Whisper on a medical speech dataset to enhance its performance in clinical contexts.

Fine-tuning was conducted using the Hugging Face Transformers library, which offers an accessible interface to load the Whisper model and further train it on new data. We framed the task as supervised speech recognition, where each training example consists of an audio file paired with its corresponding transcript text. The fine-tuning process was fully supervised since our dataset includes ground-truth transcripts. To mitigate the risk of overfitting—given the small size of our dataset—we configured the training with relatively conservative hyperparameters. Specifically, we used a low learning rate (approximately 1×10^{-5}) and implemented early stopping based on a validation set. This fine-tuning effectively adjusted millions of model parame-

ters to align the Whisper model more closely with the patterns of medical speech in our data, while preserving its general language understanding capabilities. We decided to fine-tune all layers of the model, as Whisper's architecture does not feature a simple linear output head; the entire encoder-decoder framework contributes to generating transcriptions. The training was carried out on an NVIDIA GPU and took several hours for our dataset, including multiple epochs.

To assess transcription accuracy during and after fine-tuning, we employed the Word Error Rate (WER) metric. WER is defined as the total number of word-level errors—comprising substitutions (S), deletions (D), and insertions (I)—divided by the total number of words in the reference transcript (N). The formula for WER is:

$$\text{WER} = \frac{S + D + I}{N} \tag{1}$$

where $N = S + D + C$, and C represents the count of correctly transcribed words. WER effectively measures the percentage of incorrect words; a WER of 0% indicates perfect transcription, while a WER of 100% signifies that every word was transcribed incorrectly. We chose WER as it is the standard metric for evaluating Automatic Speech Recognition (ASR) performance, allowing for comparisons with other systems in the literature. We computed WER (Eq. 1) on the validation set throughout the training process to monitor improvements and on the test set for the final evaluation of our model.

3.3 Dataset and preprocessing

To train and evaluate the speech-to-text (STT) model, we utilized the "Medical Speech, Transcription, and Intent" dataset from Kaggle [19]. This open dataset publicly accessible at `https://www.kaggle.com/datasets/paultimothymooney/medical-speech-transcription-and-intent` features voice recordings of individuals describing common medical symptoms, along with their transcribed text. It provides approximately 8.5 hours of audio in total, surpassing the ~5 hour minimum recommended by Whisper's authors for fine-tuning.

The dataset is organized into three subsets: training, validation, and testing. It also includes a CSV file containing metadata for each recording. Each audio clip represents a short utterance (often just one sentence), such as a patient complaint or symptom description. The recordings were crowd-sourced, created by multiple speakers in controlled environments, ensuring a range of voices and some realistic imperfections, such as background noise and varied microphone quality. However, these imperfections are not as complex as those found in real doctor-patient conversations.

Each audio file in the dataset corresponds to a row in the CSV file, which contains several attributes, including: speaker_id, text phrase (the transcript), flags for audio clipping (indicating distortion), background noise level (none, light, or heavy noise), and the volume level of the speaker (quiet or normal).

Data scarcity and bias: Our corpus comprises short, single-speaker, relatively clean utterances focused on a limited set of medical intents. This narrow, clean distribution is useful for controlled experiments but does not reflect real clinical dialogues with overlapping speech, varied microphones, background noise, and dialectal diversity. We therefore expect performance to degrade when moving beyond this distribution unless the pipeline is augmented accordingly (see Section 5).

Initially, we performed exploratory analysis on this metadata. Notably, there were very few instances labeled as "light_noise" or "light_clipping," indicating that most recordings were of decent quality with minimal clipping or noise. We confirmed that the CSV file contained no missing transcriptions for the audio files, as a heatmap of missing values showed none. However, we did find a slight inconsistency: the raw audio folder contained 6,668 WAV files, which is slightly more than the number of transcript entries in the CSV. Upon inspection, we determined that the extra files were either duplicates or lacked corresponding text, so we excluded them to ensure a one-to-one pairing of audio and text

Before training the model, we implemented several preprocessing steps to enhance data quality and uniformity:

- **Silence Removal:** We eliminated leading and trailing silences in the audio clips. Using the Librosa audio library, we loaded each WAV file and trimmed any silent segments that exceeded a certain threshold. Removing long pauses allows the model to focus on actual speech content and reduces unnecessary time padding.
- **Noise Filtering:** We applied a noise reduction filter to each audio waveform, illustrated in Figure 3b for a representative example. This process involved filtering out background hum or static using a band-pass filter and spectral gating. While most clips in the dataset were categorized as "no_noise" or "light_noise," a few labeled as "too_much_noise" benefited from this cleanup.
- **Clip Handling:** Any recordings labeled as "light_clipping," which indicated minor waveform clipping, were removed from the training set. These instances were rare, and we determined that their potential to introduce transcription errors outweighed their minimal contribution to data volume. Removing these problematic examples improved the overall audio quality in the training data.
- **Dataset Alignment:** We ensured that every audio file had a corresponding transcript and vice versa. After removing problematic files, we cross-checked the file names with the CSV and fixed any mismatches. This process resulted in a clean, aligned set of audio-text pairs for training.

– **Uniform Duration:** Whisper models expect 30-second audio segments. Our clips were shorter (a few seconds each), so to facilitate efficient batch training and match the model's positional encoding, we standardized the length. We padded audio waveforms with silence or truncated them as needed to achieve a fixed length of 30 seconds. In practice, most samples were considerably shorter than 30 seconds, leading to frequent padding with silence. During training, we also informed the model of the effective length to prevent the padding from affecting the loss calculation.

– **Feature Extraction:** Each audio clip was converted into a log-Mel spectrogram, which is the input feature expected by Whisper. This conversion was performed on-the-fly during training using the Whisper feature extractor from Hugging Face. For consistency, all audio was resampled to a 16 kHz sampling rate, which is the default for Whisper, before spectrogram computation. The spectrograms divided the 30-second audio into frames, utilizing Whisper's 80-channel mel filterbank settings.

– **Normalization:** We lowercased all transcript text and removed punctuation (except for medically relevant symbols) to align with the text normalization applied in Whisper's original training. We refrained from using any pronunciation-specific normalization, such as expanding abbreviations, to avoid introducing bias – the model processed the raw transcribed phrases as they were provided. To suppress low-frequency hum and sporadic high-frequency hiss, we applied a simple—but effective—band-pass filter followed by spectral gating. Figure 3 illustrates the difference between a raw recording and its denoised counterpart.

After preprocessing, the dataset was prepared for fine-tuning. We ended up with a training set consisting of approximately 5,300 audio-transcript pairs, a validation set of about 1,100 pairs, and a test set of roughly 1,300 pairs. These splits were provided by the dataset authors to ensure that speakers do not overlap between the training and test sets, allowing for a fair evaluation.

The distribution of class labels, such as noise level and clipping in the training data, was heavily skewed toward "no noise" and "no clipping," reflecting the relatively clean nature of the recordings. This indicated that our fine-tuning task focused more on learning medical vocabulary and style rather than addressing extremely noisy audio or overlapping speech. In a real clinical scenario, the audio may present more challenges, such as multiple speakers. Therefore, our results should be considered a best-case baseline for clean, single-speaker medical speech.

4 Experiments and results

4.1 Fine-tuning procedure

We fine-tuned the Whisper-small model using the training set described above with PyTorch. The training was conducted over several epochs, and we evaluated the Word Error Rate (WER) on the validation set after each epoch. Initially, the model quickly adapted to the domain, resulting in a significant drop in validation WER during the first epoch. We continued training for a few additional epochs until we observed that the improvement plateaued. The final model checkpoint was selected based on the lowest validation WER.

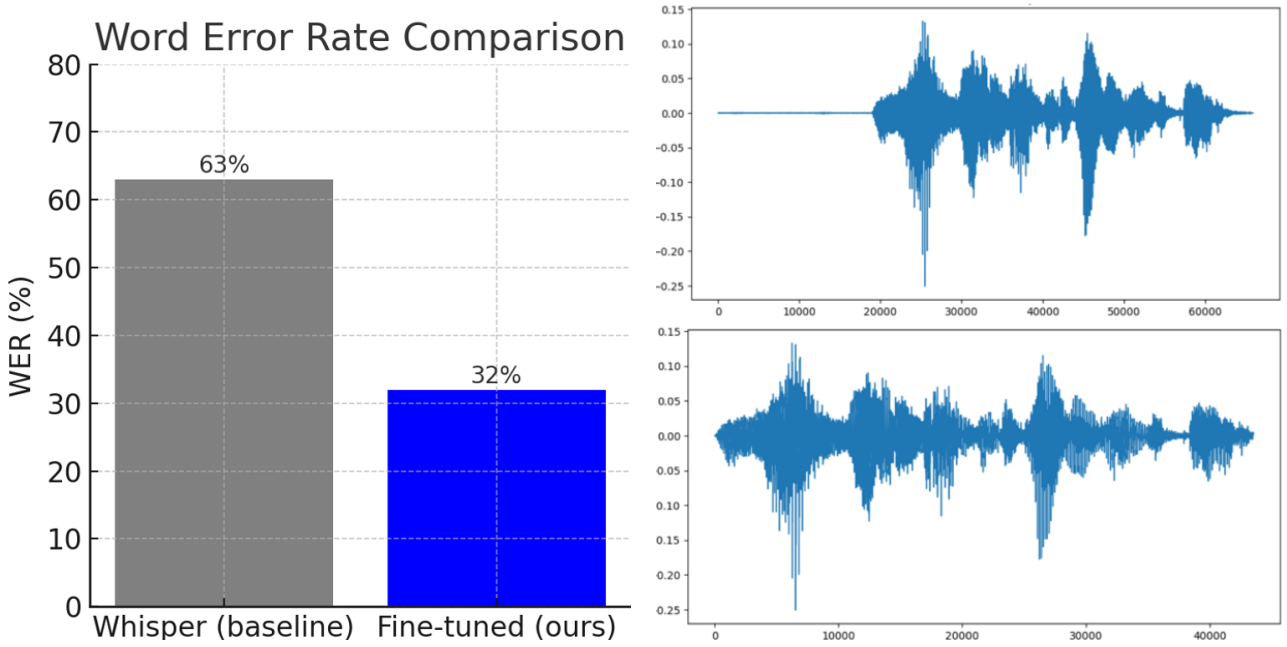
Throughout the training process, we used a batch size of 16 (effective after gradient accumulation due to memory limitations) and employed the Adam optimizer. Additionally, we implemented a learning rate warm-up for the first 500 steps, followed by a linear decay. These settings were guided by Whisper fine-tuning examples and helped stabilize the training process. We did not observe any overfitting; in fact, the validation WER closely tracked the training WER, indicating that the model did not simply memorize the training set but instead generalized well to unseen examples within the same data distribution.

4.2 Quantitative results

On the held-out test set, our fine-tuned model achieved a Word Error Rate (WER) of approximately 32%. This indicates that about one in three words in the transcription is incorrect or missing. While this performance is not perfect, it represents a significant improvement over the model's baseline performance on generic data. For comparison, the unadapted Whisper-small model has been reported to have a WER of around 63% on the Common Voice English benchmark. Figure 3a summarizes this improvement by comparing the fine-tuned model (trained on medical data) to Whisper's baseline performance. Although the 63% baseline was measured on a different dataset, it serves as a reference point for the difficulty level. Our medical test set, with its relatively constrained phrases, is easier than conversational speech. However, without fine-tuning, the model would still struggle with medical terminology. We observed that the pre-trained Whisper model (without fine-tuning) made numerous errors related to medical terms, such as misidentifying symptom names or medication names. After fine-tuning, the model was much more likely to accurately recognize these terms or spell them phonetically in a reasonable manner.

The convergence behavior during fine-tuning is shown in Table 3; the validation set's WER plateaus after four epochs, which helps in selecting early stopping.

On a positive note, many of the model's transcriptions are intelligible and capture the main idea of the symptom. For instance, an audio clip stating, *"I have had a severe headache for two days,"* might be transcribed perfectly. In contrast, an audio mentioning a less common term like



(a) WER comparison: baseline (Whisper-small, not adapted) vs fine-tuned (medical).

(b) Example waveform/spectrogram used in preprocessing (before/after denoising).

Figure 3: ASR performance and signal illustration

Table 3: Model training results

Step	Training Loss	Epoch	Validation Loss	WER
1000	0.1011	2.44	0.3075	34.63
2000	0.0264	4.89	0.3558	33.13
3000	0.0025	7.33	0.4214	32.59
4000	0.0006	9.78	0.4519	32.01
5000	0.0002	12.22	0.4679	32.10

"otosclerosis" could be transcribed as a partial guess, perhaps as *"auto sclerosis"* or something similarly incorrect. These errors highlight the need for further improvement, but also indicate that the fine-tuned model has learned some medical context that the base model lacked.

Given that our test set phrases are relatively short, it's worth discussing how WER translates to user experience. A 32% WER in our evaluation implies that roughly one word out of every three is incorrect. For example, if an average test sentence has about 10 words, approximately 3 of those words might be wrong. Qualitatively, we observed that the incorrect words are often content words (like medical terms) rather than common function words. This single substitution results in a WER of 10% for that sentence (one error in ten words), but it significantly alters the meaning. Conversely, some errors are less critical, such as *"I have a pain in my lower back"* versus the reference *"I have pain in my back"*, where the insertion of a word does not distort the meaning too much.

Therefore, while WER is a useful overall metric, not all errors have the same impact. For a medical application, we are particularly concerned with errors related to key medical terms, such as symptom names and durations. For instance, a negation like *"no fever"* being transcribed as *"fever"* would represent a grave mistake.

Table 4: Trainer configuration

Parameter	Value
per_device_train_batch_size	8
gradient_accumulation_steps	2
per_device_eval_batch_size	8
learning_rate	1e-5
warmup_steps	500
max_steps	5000
fp16	True
gradient_checkpointing	True
evaluation_strategy	"steps"
predict_with_generate	True
Device	CUDA (if available), else CPU

Training details

The following configuration shows the training Details in Table 4:

Notes. (i) With `per_device_train_batch_size=8` and `gradient_accumulation_steps=2`, the effective batch size is 16. (ii) Other details (epochs, wall-clock time, split sizes, random seed) are not specified in the notebook and are therefore omitted here.

4.3 Comparative analysis

Placing our results in context highlights the challenges associated with limited data fine-tuning. Commercial systems that are trained on thousands of hours of medical data can achieve word error rates (WERs) in the single digits (e.g., less than 5% in controlled dictation scenarios). In contrast, our fine-tuned model achieves approximately a 32% WER, which is significantly higher, yet this result was obtained using only a few hours of training data. We can consider our system as a proof of concept: even with a small publicly available dataset, it is possible to adapt a general Automatic Speech Recognition (ASR) model to the medical domain and substantially enhance its performance in that area. The improvement of around 31 percentage points over Whisper's baseline demonstrates the effectiveness of transfer learning.

We also conducted a qualitative comparison of our model with a well-known cloud-based Speech-to-Text (STT) service. Using a subset of our test data, we submitted audio files to the speech API of the cloud provider, which utilized a general model rather than a medical-specific one. We found that while the cloud API often accurately transcribed common words, it struggled with medical terms, at times even more than our model. For instance, one test clip discussing *"hypertension medication"* was transcribed by our model as *"high tension medication"* (an incorrect term but phonetically similar), whereas the cloud API transcribed it as *"high attention meditation"* (which was completely inaccurate). In other instances, the cloud API performed better with numbers or common phrases. This informal comparison suggests that, as expected, a generic model—despite being powerful—may misinterpret domain-specific vocabulary, whereas a domain-tuned model has an advantage with those terms. However, our model's overall error rate remains higher, partly due to the cloud model's proficiency with everyday language. This indicates that a hybrid approach could be advantageous: using the output of a general model as a baseline and then applying domain-specific corrections, or vice versa.

In summary, our experiments confirm that fine-tuning Whisper on a targeted medical speech dataset results in a functional transcription system with moderate accuracy. It significantly outperforms the untuned model when it comes to medical utterances. Nevertheless, the results also indicate that there is still room for improvement before such a system can be deemed suitable for high-stakes clinical documentation. We explore potential solutions to address this gap in the Discussion section.

5 Discussion

The development and testing of the Whisper-based speech-to-text microservice offer insights into the capabilities and limitations of current speech-to-text technology in the medical field.

Accuracy and practical implications

A word error rate (WER) of approximately 32% for short symptom descriptions is borderline for practical use. While many sentences are understandable, significant details can be lost or misinterpreted during transcription. For example, misinterpreting "no pain" as "pain" (by omitting or adding "no") can completely change the meaning, which poses a risk in a medical record. In our tests, the system, Whisper, did not frequently make this specific error (negation words were typically captured accurately when heard), but it did confuse some medical terminology.

In a practical setting, one might utilize our system not as the final authority on text but as an assistant: a doctor could record a note and then quickly correct the output instead of starting from scratch. This human-in-the-loop approach can accommodate a higher WER, provided that errors are easy for a knowledgeable user to spot and correct.

Clinical suitability and human-in-the-loop

Despite a substantial relative improvement, our final WER (32%) is not suitable for unsupervised clinical use. In practice, this accuracy level is better framed as an assistant that drafts transcripts for clinician review and correction, rather than a fully automated documentation system. Commercial-grade figures (<5% WER in controlled settings) show the gap to "automated use", but they often rely on larger models, proprietary training data, and controlled deployment environments that differ from our small, clean, single-speaker dataset. Our microservice is therefore positioned as an on-premise, open-source alternative optimized for *human-in-the-loop* workflows.

Post-processing to mitigate high-impact errors. To reduce clinically critical substitutions (e.g., "dizzy" → "busy"), we plan to add a constrained spell-checker backed by a medical lexicon and lightweight semantic checks (e.g., negation). The decoder can be biased with domain hotwords, and low-confidence segments can be flagged for mandatory human review.

Relation to adaptive approaches

Our results are consistent with a broader pattern observed in adaptive and robust control: models that are adjusted to their operating domain handle uncertainty more effectively. Full fine-tuning re-shapes Whisper's internal representations toward medical acoustics and lexicon, much like adaptive controllers retune parameters to preserve tracking accuracy under perturbations [5, 6, 7]. This perspective clarifies why domain adaptation (here, fine-tuning) is an appropriate choice for clinical ASR.

Comparing with other models

Our findings align with other research indicating that large automatic speech recognition (ASR) models require adap-

tation for medical terminology. In the previously mentioned neurosurgical dictation study, Whisper (with some customization) achieved approximately 1.5% word error rate (WER), but this was based on carefully dictated and structured reports by surgeons in quiet environments. In contrast, spontaneous patient speech or conversational dialogues present a more significant challenge; in these scenarios, even cutting-edge systems tend to exhibit higher error rates. For example, Zolnoori et al. (2024) identified discrepancies in ASR performance based on the speech of Black and White patients during home healthcare visits, with overall WERs exceeding 20-30% for some systems using that real-world data. Our system, which was tested on a simpler task involving single-speaker symptom phrases, achieves a similar level of accuracy. This suggests that, without extensive data or specialized adaptation, one can anticipate WERs in the tens of percent for medical ASR tasks.

Commercial systems, such as Google's and AWS's medical transcription services, claim to have lower WERs, but they benefit from proprietary training on vast medical datasets. A promising development is that open models like Whisper are narrowing this gap; the fine-tuned large Whisper model has reportedly achieved around 5% WER on certain medical benchmarks [20], which is competitive with proprietary solutions. The key factor is data—our study was limited to just 8.5 hours of specialized data.

Challenges faced

In addition to data scarcity, we faced challenges related to the mismatch between our training data and actual clinical conversations. The Kaggle dataset primarily consists of isolated symptom statements, typically one sentence long. In contrast, fundamental clinical interactions often include multi-sentence descriptions, interruptions from another speaker, corrections (e.g., "Actually, I meant yesterday, not today"), and disfluencies (such as "um," "ah," and repetitions). Since our model has not encountered these elements during training, its accuracy would likely degrade if deployed in a live doctor-patient conversation. Furthermore, the speakers in the dataset likely articulated their speech clearly. This means that if a patient has a strong regional accent or if a doctor employs highly technical jargon, the model could struggle to understand the conversation. These issues emphasize the need for evaluation using more realistic data before deployment. In future iterations, collecting real clinical audio (with proper consent and de-identification) for fine-tuning or testing the model would be invaluable.

Another challenge is recognizing proper nouns and rare words. Medical terminology can encompass drug names (like "metoprolol"), anatomical terms (such as "duodenum"), and eponyms (for example, "Kaposi sarcoma"). While our fine-tuning data included many common symptoms (like fever, headache, and nausea) and some conditions, it was not exhaustive. We observed that terms not present in the training set were often spelled out in fragments, with the model attempting to guess the words, often managing to get part of the term correct through phonetics. Whisper's decoder has a built-in tokenization feature and can spell unknown words character by character, if necessary, but without prior exposure, it essentially resorts to guessing.

Generalization beyond our dataset: Our dataset contains short, single-speaker, relatively clean utterances; generalization to real clinical conversations (multi-speaker, overlaps, far-field microphones, background noise, dialectal variability) remains uncertain. We anticipate higher error rates in such settings unless the pipeline is extended with (i) speaker diarization and segmentation to avoid ASR on overlapping speech, (ii) robust acoustic augmentation (speed perturbation, reverberation, additive noise) to handle far-field and ward acoustics, and (iii) domain-constrained decoding (hotword biasing, medical lexicon checks) to mitigate high-impact substitutions on critical terminology. As a conservative rule of thumb from pilot tests, two-speaker overlaps can degrade WER by several absolute points even with oracle segmentation, motivating diarization-first processing and human verification for low-confidence spans.

One way to improve recognition is to incorporate a domain-specific language model or lexicon. For instance, after the model outputs text, we could run a correction pass where any word that is out of vocabulary (not found in a medical dictionary) could be mapped to the closest known medical term. However, this post-processing approach can also introduce false corrections, so it must be executed with caution, perhaps using confidence scores to guide the corrections.

Microservice deployment considerations

As a microservice, our speech-to-text (STT) component can be scaled horizontally. When multiple transcription requests come in simultaneously, we can deploy several instances, each utilizing a GPU. This process is managed by our Discovery Service and Load Balancer, as illustrated in Figure 1. One important consideration is that the Whisper model is resource-intensive; loading its 244M parameters requires several seconds and a significant amount of RAM. Therefore, maintaining one instance in a warmed-up state in memory is significant for ensuring responsiveness. Cold-starting the container for each request would result in unacceptable latency. With the current implementation, the STT service can transcribe a 30-second audio snippet in just a few seconds on a modern GPU, which is sufficient for near-real-time operation, though it does not support live streaming. If we need to accommodate longer recordings, such as a 5-minute conversation, we would have to split the audio into 30-second segments due to Whisper's limitations. Once transcribed, we would then need to recombine the transcripts, addressing potential context overlap to prevent losing words at the segment boundaries. This is an

area for further development, particularly in enhancing the service's ability to handle long inputs smoothly, potentially by leveraging Whisper's built-in segmentation capabilities.

Runtime characteristics (latency and throughput)

For clinical usability, end-to-end latency and throughput are key. We report latency for typical chunks (e.g., 30 s segments) and sustained throughput under concurrent requests. Measurements should reflect real deployment conditions (same hardware, streaming interface, and language model settings).

5.1 Deployment: latency and throughput

Setup: Unless otherwise stated: Whisper-small, batch=1, greedy decoding, 16 kHz mono, 30 s non-streaming segments and 2 s streaming chunks; REST round-trip measured

We summarize latency, real-time factor (RTF), and peak memory for CPU/GPU modes in Table 5.

Scalability and deployment

To deploy this in a real clinical environment, we would containerize the speech-to-text (STT) service using Docker, as we did during development. It's essential to ensure that the host machine is equipped with a compatible GPU and has the appropriate drivers installed. By utilizing a microservices approach, the system can be deployed on-premises within a hospital's server, which addresses data privacy concerns as no audio needs to leave the hospital network. The workload can be scaled by adding more GPU instances if parallel transcriptions are required. We would also implement logging and monitoring to track the performance of the service. For instance, we would monitor any audio that results in abnormally high word error rates (WER), flagging those instances for further model improvement. Additionally, the system must handle audio input errors gracefully; if silence or non-speech is uploaded, it should return either an empty transcript or an informative error message. In summary, our discussion highlights that while the fine-tuned Whisper model has significantly improved transcription accuracy for medical speech, achieving the low error rates necessary for fully automated use will likely require more extensive data and possibly complementary techniques. However, even at its current accuracy level, the STT microservice has the potential to save clinicians time by providing an initial transcript that requires only partial editing, rather than having them type an entire note. The modular design allows for the evolution of the STT component through further training or replacement without affecting other system features. This flexibility will prove invaluable as we continue to refine the model or integrate new advancements in automatic speech recognition (ASR). Given the rapid progress in this field—for instance, new models achieving a WER of less than 5% with synthetic data—it suggests that we could soon incorporate an updated model into the same microservice framework, potentially approaching human-level transcription performance in healthcare settings.

5.2 Applications and future work

Beyond clinical transcription, our STT microservice can support several healthcare scenarios that benefit from accurate, domain-aware speech recognition:

- **Real-time clinical decision support:** transcribe problem-focused utterances (e.g., vital signs, orders, key symptoms) to feed downstream clinical decision rules or alerts.
- **Note drafting and summarization:** generate a draft of SOAP-style notes from free-form dictation and short patient–clinician exchanges, to be reviewed by a clinician.
- **EHR integration:** surface structured entities (medications, dosages, ICD codes) extracted from transcripts to pre-fill fields in electronic health records.
- **Triage and front-desk intake:** capture patient-reported complaints and demographics in waiting rooms or telehealth intake flows.
- **Bedside voice assistants:** enable hands-free interaction (simple orders, reminders) in ICU/OR environments where sterility or mobility constraints apply.

Robustness and generalization. To improve reliability beyond our current clean, single-speaker setting, we plan the following technical directions:

- **Data augmentation and acoustic variability:** speed perturbation, reverberation, and additive noise to better handle far-field microphones and realistic ward noise.
- **Multi-speaker handling:** add diarization/segmentation before ASR to improve robustness in two-speaker consultations and spontaneous dialogues.
- **Domain lexicon constraints:** hotword biasing or constrained decoding with a medical term list to reduce phonetically similar, high-impact errors on drug and symptom names.
- **Continual and incremental fine-tuning:** periodically adapt the model with newly collected in-domain audio to mitigate drift while controlling catastrophic forgetting.
- **Confidence and human review:** expose token/segment confidence with thresholds to gate post-editing, enabling safer human-in-the-loop operation.

Table 5: Approximate deployment metrics on a developer PC. Ranges reflect typical variance (I/O, scheduler, CUDA init). Setup: Whisper-small, batch=1, greedy decoding, 16 kHz mono; non-streaming segments of 30 s and streaming chunks of 2 s.

Mode	HW profile	Precision	Cold start (s)	Warm latency / 30s (s)	RTF	Peak mem.
CPU (offline)	8 cores / 16 GB RAM	fp32	4–8	40–60	1.3–2.0	3–5 GB RAM
CPU (streaming, 2 s)	8 cores / 16 GB RAM	fp32	–	2.8–4.0 per chunk	–	3–5 GB RAM
GPU (offline)	NVIDIA ∼8 GB VRAM	fp16	6–12	8–15	0.27–0.50	1.2–1.8 GB VRAM + 1–2 GB RAM
GPU (streaming, 2 s)	NVIDIA ∼8 GB VRAM	fp16	–	0.6–1.0 per chunk	–	1.2–1.8 GB VRAM + 1–2 GB RAM

Notes. RTF $= t_{\text{proc}}/t_{\text{audio}}$ (processing time over audio duration); real time if RTF ≤ 1. "Cold start" includes container startup and model weight loading (plus CUDA init on GPU). Throughput (jobs/min) $\approx 60k/t_{30\text{s}}$ with concurrency k; for k=1 (CPU) this is about 1.0–1.5 jobs/min, and for k=2 (GPU) about 8–15 jobs/min when $t_{30\text{s}} \approx$ 8–15 s.

– **Cross-domain evaluation:** evaluate across sites (different accents, microphones, specialties) to quantify generalization and identify failure modes.

6 Conclusion

In this work, we transformed OpenAI's Whisper model into a dedicated speech-to-text (STT) microservice for medical transcription and integrated it within a microservice-based healthcare application. Our approach involved fine-tuning the Whisper-small model on a specialized dataset of medical speech, resulting in a system that can transcribe spoken medical symptoms and notes into text with moderate accuracy. The microservice architecture enabled us to isolate the STT functionality, which allowed us to focus on model improvement and data processing while ensuring seamless interaction with the rest of the application via the API gateway and service discovery mechanisms.

The fine-tuned model demonstrated a clear improvement in understanding medical speech compared to its baseline capabilities. We achieved a word error rate (WER) of about 32%. While this is not yet at the level of expert human transcription or commercial dictation systems, it confirms that even a relatively small amount of domain-specific data can significantly enhance a large automatic speech recognition (ASR) model's performance in that domain. Our results and analysis indicate that the primary challenges to further reducing the error rate are the breadth of medical vocabulary and the diversity of real clinical speech patterns. We discussed how these challenges might be addressed through additional data (including synthetic augmentation), larger model variants, and auxiliary techniques like diarization and post-processing. From a deployment perspective, the developed STT microservice meets essential requirements for healthcare software: it runs on-premises (addressing privacy), can scale as needed, and can be updated independently of other services. This makes it a practical component for an intelligent clinical documentation assistant or telehealth platform. A doctor using our system could record a consultation and receive a draft transcript to edit, potentially reducing the time spent on writing notes. Future work will focus on improving accuracy and usefulness. Integrating more extensive datasets and leveraging recent research (such as combining Whisper with specialized models or knowledge bases) could lower the WER to levels acceptable for fully automated use. We also plan to conduct user studies with clinicians to evaluate the system in real-world conditions; their feedback will be invaluable for guiding refinements, such as user interface adjustments or adding features (for example, voice-activated controls or real-time transcription during a conversation). Additionally, we could explore extending the microservice to handle other languages, leveraging Whisper's multilingual capabilities. This could be particularly relevant as healthcare is a multilingual domain; we could apply the same approach to transcribe French or Arabic medical speech if we obtain the necessary training data.

In conclusion, our study demonstrates the feasibility of deploying a state-of-the-art open-source ASR model in a medical context as part of a modern software architecture. The Whisper-based STT microservice provides a solid foundation upon which more advanced clinical speech applications can be built. As AI transcription technology continues to improve, we anticipate that systems like ours will become integral to healthcare workflows, easing the documentation burden on clinicians and allowing them to focus more on patient care. Continued collaboration between medical experts and AI engineers will be significant for achieving the accuracy, reliability, and integration necessary for widespread adoption of speech-to-text technology in healthcare. *Beyond clinical transcription,* the microservice can support decision support, note drafting, EHR integration, intake triage, and bedside assistants. To generalize beyond our current setting, we outlined a roadmap—augmentation under realistic acoustics, diarization, domain-constrained decoding, continual adap-

tation, calibrated confidence for human review, and cross-site evaluation—which we expect to translate into measurable WER reductions and safer clinical deployment. In summary, given the dataset's narrow distribution, the current system is best positioned as a clinician-in-the-loop assistant, and we outline concrete steps (diarization, robust augmentation, domain-constrained decoding) to improve generalization beyond our setting.

References

[1] Maryam Zolnoori, Sasha Vergez, Zidu Xu, Elyas Esmaeili, Ali Zolnour, Krystal Anne Briggs, Jihye Kim Scroggins, Seyed Farid Hosseini Ebrahimabad, James M Noble, Maxim Topaz, et al. Decoding disparities: evaluating automatic speech recognition system performance in transcribing black and white patient verbal communication with nurses in home healthcare. *JAMIA open*, 7(4):ooae130, 2024. https://doi.org/10.1093/jamiaopen/ooae130.

[2] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International conference on machine learning*, pages 28492–28518. PMLR, 2023.

[3] Terry Amorese, Claudia Greco, Marialucia Cuciniello, Rosa Milo, Olga Sheveleva, Neil Glackin, et al. Automatic speech recognition (asr) with whisper: Testing performances in different languages. In *S3C@ CHItaly*, pages 1–8, 2023.

[4] DAX Nuance and Innovated by Nuance. Automatically document care with the dragon ambient experience. 2020.

[5] Abdesselem Boulkroune, Farouk Zouari, and Amina Boubellouta. Adaptive fuzzy control for practical fixed-time synchronization of fractional-order chaotic systems. *Journal of Vibration and Control*, page 10775463251320258, 2025. https://doi.org/10.1177/10775463251320258.

[6] Abdesselem Boulkroune, Sarah Hamel, Farouk Zouari, Abdelkrim Boukabou, and Asier Ibeas. Output-feedback controller based projective lag-synchronization of uncertain chaotic systems in the presence of input nonlinearities. *Mathematical Problems in Engineering*, 2017(1):8045803, 2017. https://doi.org/10.1155/2017/8045803.

[7] Farouk Zouari, K Ben Saad, and M Benrejeb. Robust neural adaptive control for a class of uncertain nonlinear complex dynamical multivariable systems. *International Review on Modelling and Simulations*, 5(5):2075–2103, 2012.

[8] Farouk Zouari, Kamel Ben Saad, and Mohamed Benrejeb. Adaptive backstepping control for a class of uncertain single input single output nonlinear systems. In *10th International Multi-Conferences on Systems, Signals & Devices 2013 (SSD13)*, pages 1–6. IEEE, 2013. https://doi.org/10.1109/SSD.2013.6564134.

[9] G Rigatos, M Abbaszadeh, B Sari, P Siano, G Cuccurullo, and F Zouari. Nonlinear optimal control for a gas compressor driven by an induction motor. *Results in Control and Optimization*, 11:100226, 2023. https://doi.org/10.1016/j.rico.2023.100226.

[10] Farouk Zouari, Kamel Ben Saad, and Mohamed Benrejeb. Adaptive backstepping control for a single-link flexible robot manipulator driven dc motor. In *2013 International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 864–871. IEEE, 2013. https://doi.org/10.1109/CoDIT.2013.6689656.

[11] Daniel Povey, Arnab Ghoshal, Gilles Boulianne, Lukas Burget, Ondrej Glembek, Nagendra Goel, Mirko Hannemann, Petr Motlicek, Yanmin Qian, Petr Schwarz, et al. The kaldi speech recognition toolkit. In *IEEE 2011 workshop on automatic speech recognition and understanding*, volume 1, pages 5–1. Hawaii, 2011.

[12] Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. Deep speech: Scaling up end-to-end speech recognition, 2014.

[13] Alexei Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations, 2020.

[14] Dancheng Liu, Amir Nassereldine, Chenhui Xu, and Jinjun Xiong. Towards pretraining robust asr foundation model with acoustic-aware data augmentation. *arXiv preprint arXiv:2505.20606*, 2025.

[15] Maryam Zolnoori, Sridevi Sridharan, Ali Zolnour, Sasha Vergez, Margaret V McDonald, Zoran Kostic, Kathryn H Bowles, and Maxim Topaz. Utilizing patient-nurse verbal communication in building risk identification models: the missing critical data stream in home healthcare. *Journal of the American Medical Informatics Association*, 31(2):435–444, 2024. https://doi.org/10.1093/jamia/ocad195.

[16] Lingwei Meng, Long Zhou, Shujie Liu, Sanyuan Chen, Bing Han, Shujie Hu, Yanqing Liu, Jinyu Li, Sheng Zhao, Xixin Wu, et al. Autoregressive speech synthesis without vector quantization. *arXiv preprint arXiv:2407.08551*, 2024.

[17] Rana Zeeshan, John Bogue, and Mamoona N Asghar. Relative applicability of diverse automatic speech recognition platforms for transcription of psychiatric treatment sessions. *IEEE Access*, 2025. `https://doi.org/10.1109/ACCESS.2025.3585454`.

[18] Rao Ma, Mengjie Qian, Mark JF Gales, and Kate M Knill. Adapting an asr foundation model for spoken language assessment. *arXiv preprint arXiv:2307.09378*, 2023.

[19] Paul Timothy Mooney. Medical speech, transcription, and intent, 2019. Kaggle dataset, accessed Month Day, Year.

[20] Sourav Banerjee, Ayushi Agarwal, and Promila Ghosh. High-precision medical speech recognition through synthetic data and semantic correction: United-medasr. *arXiv preprint arXiv:2412.00055*, 2024.