Speech Recognition Algorithm for Detecting Mispronunciation for Research with Children

EPFL Master Thesis

Dana Kalaaji

Spring 2024

EPFL Thesis Advisor

Prof. Pierre Dillenbourg

FCBG Supervisor

Nathan Attia

EPFL Supervisor

Lucas Burget

Acknowledgments

This report concludes my master thesis conducted at the Virtual Reality (VR) Platform within the Fondation Campus Biotech Geneva (FCBG). Established in 2013, FCBG is a non-profit foundation formed through a collaboration between the Ecole Polytechnique Fédérale de Lausanne (EPFL), the University of Geneva (UNIGE), and the Canton of Geneva. FCBG's mission is to support interdisciplinary research in fields such as neuroscience, bioengineering, and digital health by providing advanced infrastructure and expertise. This project represents a collaboration between FCBG's VR Platform, which focuses on immersive interaction and motion analysis for experimental research and clinical applications, and the Swiss National Centre of Competence in Research (NCCR) Evolving Language, specifically Professor Daphné Bavelier's research group at the University of Geneva.

My heartfelt thanks go to the HNP VR platform team at FCBG: Nathan Attia, Maël Lacour, Margaux Dubessy, Arthur Trivier, Antoine Provost, and Hugo Farajallah. Their warm welcome and the enriching experiences we shared have been invaluable. I am especially indebted to Nathan Attia, my supervisor, whose guidance, mentorship, and unwavering support were crucial to the success of this project.

I am deeply grateful to Professor Daphne Bavelier and Professor Marina Laganaro, along with their respective teams, for their insightful direction and constructive feedback. Their expertise significantly shaped the course of this work. In particular, I would like to thank Jessy, Angela, Manon, and Julie for the efforts they put into helping me with the data, and for their quick responsiveness.

From EPFL, my sincere appreciation extends to Professor Pierre Dillenbourg for his insightful supervision and valuable feedback throughout this project. I am particularly grateful to Lucas Burget, PhD at from the Chilli lab led by Professor Dillenbourg, whose knowledge and hands-on guidance were truly invaluable in this research.

Finally, I acknowledge the helpful support provided by Enno and Mathew from the Idiap Research Institute and Nianlong and Guanghao from the Zurich NCCR TTF research institute. Their expertise have been of great help in the development of this master's thesis

To all mentioned here, and those whose names may not appear but have contributed in various ways, I extend my heartfelt thanks for your support and encouragement throughout this academic journey.

Contents

1 Introduction

Automatic Speech Recognition (ASR) technology, which converts spoken language into text, has rapidly advanced, playing a crucial role in enhancing human-machine interactions. Traditional ASR systems, developed in the 1970s, relies on a pipeline of separate components each trained independently. The rise of end-to-end ASR systems represents a significant shift in the field. Unlike traditional ASR, end-to-end ASR integrates these components into a single neural network, simplifying the pipeline and often resulting in more efficient training and improved performance as they learn to map audio signals directly to text sequences.

Building upon recent advancements in ASR, there is a promising opportunity to apply this technology to educational contexts, particularly in literacy development. The latter is a critical aspect of early education, with particular importance for children experiencing reading difficulties. Traditional diagnostic methods for assessing reading skills often rely on labor-intensive and time-consuming manual processes that involve detailed transcriptions and analysis of children's reading performances. However, ASR models are primarily trained on adult speech data, presenting unique challenges when applied to children's speech patterns and pronunciations.

To address these challenges and bridge the gap between ASR technology and educational needs, this master thesis proposes the development of a real-time model capable to accurately determine the correctness of spoken words or non-words from predefined lists given to children. It's important to note that this project focuses on classifying whether an item is pronounced correctly or not, rather than attempting to detect or transcribe what was actually said. The primary objective of this research is to create a model that can be seamlessly integrated into the speech evaluation pipeline used by educators and clinicians. By automating the process of detecting mispronunciations, we aim to alleviate the mechanical aspects of assessment from the professionals, thereby allowing them to focus their expertise on analyzing and addressing the detected mispronunciations.

2 Child Speech Recognition: Challenges and Models

2.1 Unique Challenges of Child Speech Recognition

ASR faces unique challenges when processing children's speech, primarily due to acoustic and pronunciation variability and insufficient child-specific speech datasets:

- **Physiological differences:** Children exhibit distinct spectral and temporal speech patterns compared to adults due to their smaller vocal tracts and developing articulatory mechanisms [1, 2]. Firstly, the fundamental frequency of children's voices can reach up to 400 Hz, significantly higher than that of adult males (around 120 Hz) and females (around 220 Hz) . The smaller vocal tract size in children results in shifted formant frequencies, which are crucial for phoneme distinction in ASR systems [3]. Furthermore, children's speech patterns differ from adults' in other various aspects, including a higher incidence of mispronunciations, more frequent disfluencies, a greater presence of non-verbal vocalizations, and a lower articulation rate. They also exhibit a higher duration of pauses in their speech compared to adults, often using these breaks less strategically. These variations pose significant challenges for ASR systems designed primarily for adult speech [4, 5, 6, 7].

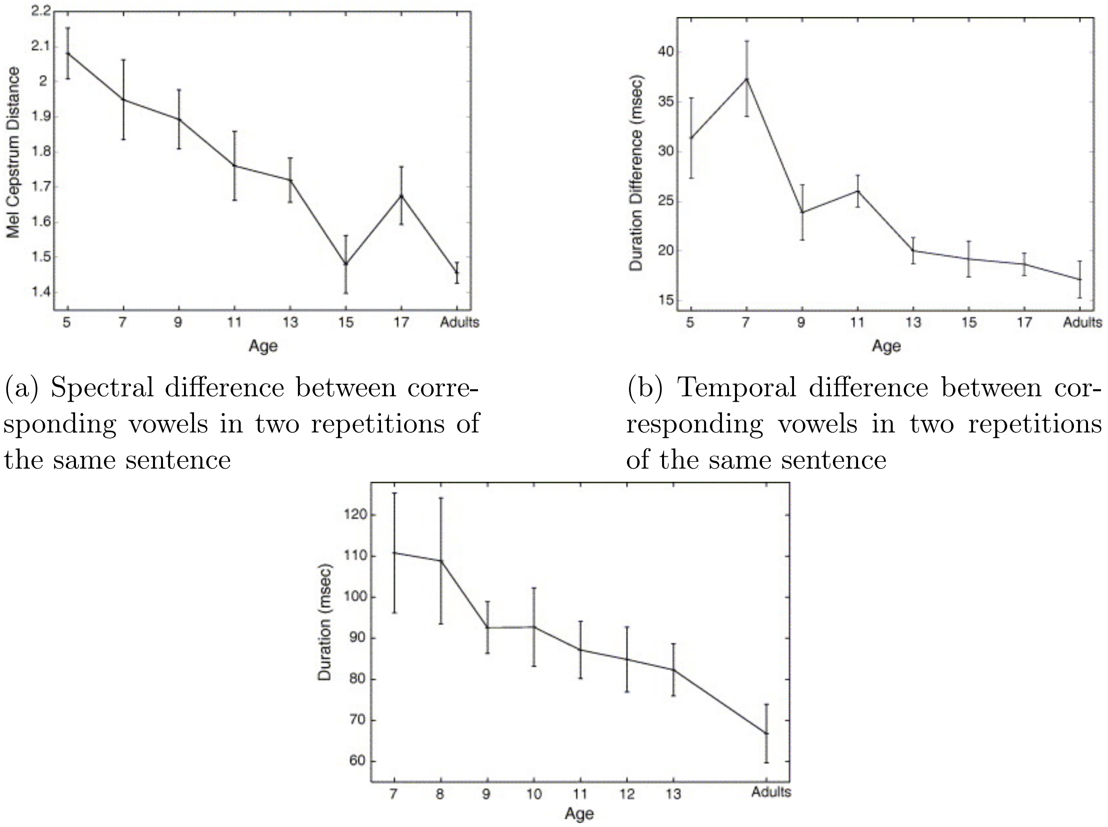
- **Increased variability:** The developmental stage of children introduces greater inter and intra variability in their speech patterns compared to adults [6, 2]. Gerosa et al. [8] demonstrated that Vocal Tract Length Normalization (VTLN), aimed at reducing speaker variability in the frequency domain, improved speech recognition for both adults and children, but its lower effectiveness for children (5.3% error rate reduction compared to 10.5% for adults) highlights the impact of factors beyond vocal tract length, such as intra-speaker variability, in the challenges of recognizing children's speech. Moreover, their study [5] conducted on english and italian child speech revealed several key findings that they illustrated in Figure 1:

  - The mean phone duration decreases with age and is significantly higher for children than for adults.
  - The variability within the same age group decreases as children get older, indicating greater inter-speaker differences in younger children.
  - Intra speaker variability decreases with age both in spectral and temporal aspects.

  These inconsistencies in speech patterns present substantial challenges for Automatic Speech Recognition (ASR) systems, particularly in maintaining consistent accuracy across different age groups, with younger speakers posing the greatest difficulty. A study by Shivakumar and Georgiou [9] highlights the challenge of improving ASR for younger speakers, indicating that only increasing the training data may not suffice to close the performance gap as they found that:

  - Higher variability in younger children's speech necessitates significantly more data to adapt adult ASR systems to this age group. For older children, only 30 minutes of adaptation data is needed to outperform an ASR trained on 90 hours tested on younger children.
  - Despite using large amounts of age-matched adaptation data, ASR performance for children aged 6-8 years consistently falls short of the accuracy achieved for older age groups.

- **Data scarcity:** There is a lack of datasets for children's speech, which limits the ability of ASR systems to accurately transcribe and understand the children's speech which, as aforementioned, is different from adult speech. This scarcity is particularly acute for languages other than English and for specific age groups.



(a) Spectral difference between corresponding vowels in two repetitions of the same sentence

(b) Temporal difference between corresponding vowels in two repetitions of the same sentence

(c) Mean phone duration for children of different ages and adults

Figure 1: Intra [(a),(b)] and Inter (c) speaker differences in phoneme variation (from [5])

2.2 End-to-End Models in Child Speech Recognition

Recent studies have shown that end-to-end models outperform traditional ASR systems in speech recognition tasks, including those featuring children [7, 10, 11]. CNN-based end-to-end acoustic models learn to model formant information invariant to the acoustic differences in children and adult speech [12]. Despite this, end-to-end ASR systems trained on adult speech often struggle with children's speech, exhibiting error rates 10 to 19 times higher.[7]. Even with adaptation, children's ASR performance remains more than 6 times worse than adult ASR. These findings highlights the presence of other unique characteristics in child speech from both adult speech that still require specific adaptation. For our preliminary evaluation on our data, we selected two state-of-the-art ASR models: WhisperX and Wav2vec2.

2.2.1 WhisperX

Whisper [13], developed by OpenAI, is a general-purpose speech recognition model. It uses a fully supervised end-to-end architecture based on an encoder-decoder Transformer. The model was trained on 680,000 hours of diverse audio data, including 65% English, 18% non-English to English translation, and 17% non-English in 98 different languages with multiple accents. Whisper key advantages are the robustness and zero-shot performance across diverse datasets, achieving near state-of-the-art accuracy in speech recognition across a wide range of scenarios and technical language [14, 13], It is particularly robust with accents, dialects, and background noise and demonstrates superior performance in uncontrolled acoustic conditions, and overall outperforms Wav2Vec2 with smaller performance gaps between different speaker groups [15, 14]. For our evaluation, we will use *WhisperX* [16] (see Figure 2), an extension of Whisper. WhisperX improves upon its predecessor by running in real-time, reducing hallucination in transcriptions and returning more accurate word timestamps.

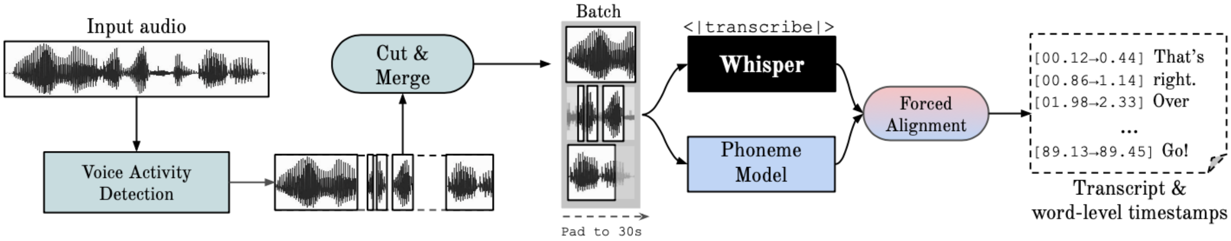


Figure 2: The WhisperX Model

2.2.2 Wav2vec2

Wav2vec2 [17] (see Figure 3), introduced by Facebook AI Research in 2020, is a self-supervised learning framework for speech recognition. Its architecture includes a feature encoder, a Transformer network for contextualized representations, and a quantization module. The model undergoes a two-phase training process: it is first pre-trained on extensive amounts of unlabeled audio data to learn general speech patterns, then finetuned on labeled data by adding a randomly initialized output layer on top of the Transformer to maps the learned representations to text. This approach allows Wav2vec2 to offer several benefits:

- Adaptability: The models are pre-trained on a large quantity of unlabeled data and can be fine-tuned to specific scenarios with relatively small amounts of speech data. This is particularly beneficial for scenarios with limited resources, such as child speech recognition [18, 10, 11]. Studies demonstrated that fine-tuning a Wav2vec2.0 model pre-trained on adult speech with only a few hours of child speech data outperforms the baseline model finetuned on 960h of adult speech [18, 19]. Moreover, while underperforming in general compared to Whisper, Wav2vec2 outperforms in child speech recognition when fine-tuned [20, 21].

- LM Independence: (See Table 1 for definition) Wav2vec2 can operate without a language model. In such cases, the model relies primarily on acoustic input rather than contextual language modeling, which can potentially be beneficial for transcribing pseudowords or detecting mispronunciations.

For our evaluation, we tested multiple fine-tuned models on adult French ASR:

- *VOX* [22]: large model pretrained on 100k hours of unlabeled VoxPopuli dataset (parliament speech) across 23 languages and finetuned on the french Common Voice 7.0 dataset (read speech)

- *XLS* [23]: multilingual model XLS-R [24] pretrained on 500,000 hours of audio across 128 languages and the datasets shown in Figure 4 and finetuned on multiple datasets (read speech datasets among pretraining and finetuning datasets)
- *Phonemizer* [25]: base model pretrained on 22.8k hours of unlabeled VoxPopuli French dataset (parliament speech) and finetuned for French speech-to-phoneme using the Common Voice 13 dataset (read speech)

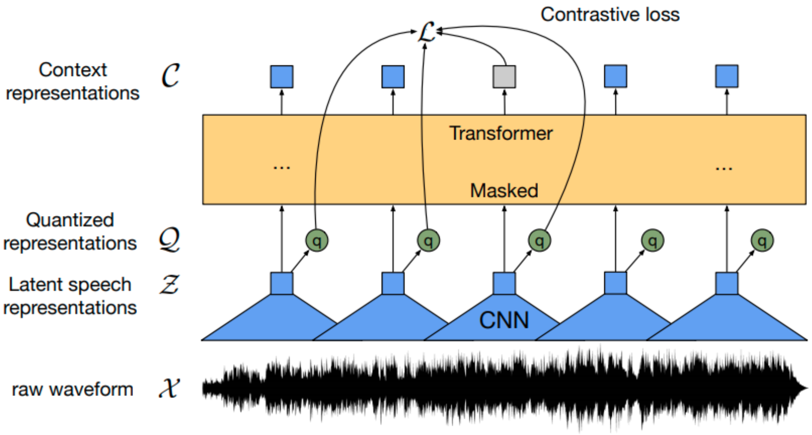

Contrastive loss
$\mathcal{L}$
Context representations $\mathcal{C}$
Transformer
...
Masked
Quantized representations $\mathcal{Q}$
q
Latent speech representations $\mathcal{Z}$
CNN
raw waveform $\mathcal{X}$

Figure 3: The Wav2vec2 Model

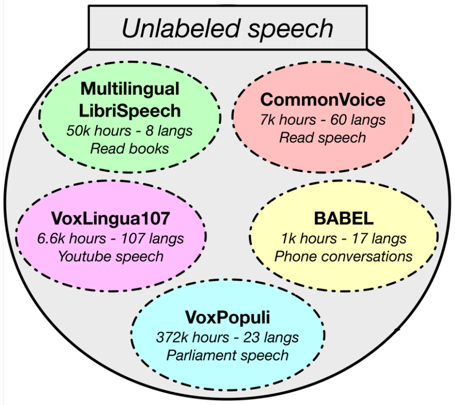


Figure 4: XLS Pretraining Datasets. Vox model was only pretrained on VoxPopuli.

Language Model (LM): Statistical model that estimates the likelihood of word sequences in a language, improving transcription accuracy by favoring more probable word sequences.
Hotwords: Words given higher priority in the recognition process.

Table 1: Concepts Definition

3 Data

3.1 Reading Task

This study focuses on one common reading assessment task, namely the decoding task. It was designed to evaluate children's reading skills and is particularly important for assessing early literacy development and identifying potential reading difficulties in young children. The decoding task requires children to read aloud lists of isolated words and pseudo words as quickly and accurately as possible.

The task comprises 18 different lists, referred to as **'Config IDs'**. Each list consists of 12 elements, either words or pseudowords, totalling 96 items across all lists. We call **'Reference Text'** the content of each list. The 18 lists are divided into three groups (A, B, C) of comparable difficulty. Each group contains six distinct lists, categorized as follows:

- Two Easy Lists: common words with simple phonetic structures.

`nuit métal joue escalade valet tente jaloux couleur fossé noix balade reptile`

- Two Complex Lists: less frequent words or with more intricate phonetic structures.

`piège femme secret finit mille cerf jardin précision dix lieux million débarquement`

- Two Pseudo Lists: non-words that adhere to French phonological rules.

`suf fari juit lumèce goix munon donte tondé toir rombage jeur brète`

A total of 155 French-speaking participants between 7 and 8 years old completed the decoding task, each one was asked to read all 6 lists from one group. The total audio duration for the decoding dataset is around 6.5 hours. Table 2 provides a detailed breakdown of the number of participants and total audio length for each Config ID.

3.2 Audio Quality

The audio quality varies across the dataset. Some recordings contain background noise and microphone buzz is often present. In some instances children are heard whispering, and many recordings include disfluencies such as false starts, insertions repetitions, or prolongations. Moreover some recordings are partially cut, partially or fully inaudible, or challenging even for human listeners to comprehend. It's important to note that the speech in this dataset is read speech, not spontaneous speech. This characteristic adds to specific patterns in the recordings [26, 27, 28]. Some children make long pauses within words or between words, while others may read two words very quickly and then pause between syllables of the same word. All these factors, combined with mispronunciations, the presence of pseudo-words, and the challenges in children's speech recognition can make it harder for the ASR to accurately transcribe the audio.

Configuration	Num Participants	Duration (minutes)
config_A_easy_1	47	17
config_A_easy_2	47	18
config_A_complex_1	47	17
config_A_complex_2	47	22
config_A_pseudo_1	47	18
config_A_pseudo_2	47	22
config_B_easy_1	55	25
config_B_easy_2	55	25
config_B_complex_1	55	26
config_B_complex_2	55	27
config_B_pseudo_1	55	22
config_B_pseudo_2	55	27
config_C_easy_1	53	21
config_C_easy_2	53	20
config_C_complex_1	53	23
config_C_complex_2	53	28
config_C_pseudo_1	53	21
config_C_pseudo_2	53	26

Table 2: Number of participants and minutes of data per Configuration ID

3.3 Transcription and Accuracy Scoring Methodology

For each trial, a clinician annotated the child's utterance and scored the pronunciation accuracy following the convention among professional in speech therapy. This section outlines the used notation conventions and scoring criteria, and how the latter are mapped to a binary scoring.

3.3.1 Clinician Transcription Protocol

The transcriptions use the Latin alphabet and transcribe the sounds as pronounced by the child, rather than adhering to correct spelling to allow for identification of mispronunciations. The annotations also mark any insertions of syllables within words and instances of sounding out, and contextual notes gives external factors affecting the transcription. The following example illustrates the protocol used:

Reference:	bucurelle daveau ninoie trefonet tiège ronjeau meuil trufondulence hupeur mive gralation turlème
Transcription:	{comments} bu.turelle davo ninoi trefonet tchiètche ron ronge ronjo mail trufodulence {inaudible} dralation turlè{v}.me
Accuracy:	0 2 2 2 0 2 0 0 NA NA 0 1

Explanation of Special Notations

- **Periods ’.’ :** Indicate sounding out (or syllable separation) in the child’s utterance.
  - Example: “bu.turelle” shows that the word was pronounced with distinct syllables.
- **Curly braces ’{}’ :** Serve two purposes:
  1. To indicate insertions within words:
     - Example: “turlè{v}.me” shows both syllable separation and an inserted ’v’ sound.
  2. To provide contextual notes about the audio:
     - {comments} indicates irrelevant speech before a word.
     - {inaudible} indicates that certain words are not clearly audible.

3.3.2 Clinician Accuracy Scoring Protocol and Mapping

The clinician’s accuracy score for each utterance (child’s attempt to read aloud the given list) consists of 12 values, one for each (pseudo)word, ranging from 0 to 2. When a word is inaudible, it is marked with ’NA’. To compare with the binary correct/incorrect output of the ASR, we mapped these scores as shown in Figure 5.

Clinician’s Score	Mapping to ASR Output
2: Correct	2, 1 → Correct (1)
1: Almost correct (specific cases)	0, NA → Incorrect (0)
0: Incorrect or omission	
NA: Not applicable	

Figure 5: Clinician’s score mapping for binary classification

4 Performance Evaluation Protocol

4.1 Metric

We categorized the ASR performance by defining the following four categories:

- **True Positive (TP)**: Instances where both the ASR and clinician identified the word as correct. This indicates agreement on proper pronunciation.
- **True Negative (TN)**: Cases where both the ASR and clinician identified the word as incorrect. This represents agreement on mispronunciation, regardless of the specific transcription of the incorrect word.
- **False Positive (FP)**: Situations where the ASR identified a word as correct, but the clinician marked it as incorrect. These cases are particularly critical as they represent mispronunciations that the ASR failed to detect.

- **False Negative (FN)**: Occurrences where the ASR identified a word as incorrect, but the clinician scored it correct. These cases are particularly problematic when using the ASR as a data-driven method to determine when a task has become too difficult for a child, as they may lead to premature task termination or unnecessary interventions.

This evaluation process allows us to assess the performance of the ASR models highlighting areas where the models align with or diverge from expert human evaluation. The primary objective is to maximize the number of TP and TN while minimizing FN and FP. However the optimal balance between FP and FN depends on the specific use of the ASR:

1. **Post-hoc Analysis of Utterances:** In this scenario, the ASR serves as a first-pass evaluation tool to reduce clinicians' workload during the analysis phase by flagging items that need review. This allows clinicians to focus solely on those marked as incorrect (TN and FN), ensuring that no mispronunciations are overlooked. The key priority is to maintain a very low false positive rate, as they result in mispronunciations being incorrectly classified as correct, which causes them to bypass the clinician review step and go unnoticed. We aim to keep this rate below a realistic **5% threshold** (ideally under 1%), ensuring that clinicians can trust the ASR's correct labels. While this approach may increase the number of FN, it is acceptable because all flagged utterances will still be reviewed. Only the false positives present a risk of overlooked errors, as they do not undergo further examination by clinicians.

2. **Real-time Task Assessment:** In this scenario, the ASR serves as a real-time tool to dynamically adjust task difficulty and decide when to end data collection based on a child's performance. The primary goal is to minimize false negatives, as these occur when correct pronunciations are incorrectly flagged as incorrect, which can lead to premature termination of tasks and an inaccurate assessment of the child's abilities. By maintaining a low FN rate, we ensure that the system accurately identifies mispronunciations, allowing for reliable assessment. Although this approach may lead to a higher FP rate, it is acceptable because capturing as many true positives as possible provides a more comprehensive evaluation of the child's skills, even if it occasionally means extending tasks slightly beyond the child's optimal comfort level.

Hence, the ASR system must be designed with the flexibility to address both scenarios effectively. For the remainder of this study, we will focus on assessing performance with respect to the first case, keeping FP rate under 5%. However, the ASR should also be capable of handling the second scenario where managing FN is crucial.

4.2 Method

To evaluate the ASR model's performance, we implemented the following procedure, as illustrated in Table 3:

1. **ASR Transcription**: Process the audio data through the model to generate transcriptions.

2. **ASR Accuracy Scoring**: Compute the accuracy score by comparing each item in the ASR transcript to the reference text. Matching words received a score of 1, while mismatches are scored as 0.

3. **Clinician Performance Comparison**: Juxtapose the ASR accuracy score against the clinician one, and categorize each outcome as TP, TN, FP or FN.

Reference Text:	toit régal juin escapade navet pente bandit chaleur carré choix parade fossile
Clinician Transcription:	toi rédale join es esgabade navet pente mendi chaleur charré choix parade fossile
ASR Transcription:	toi ridal juin estabadnavé pantmendi chaleur tharéchoix parade fossiles
Clinician Accuracy:	1 0 1 0 1 1 0 1 0 1 1 1
ASR Accuracy:	0 0 1 0 0 0 0 1 0 0 1 0
Performance:	FN TN TP FN FN FN TN TP TN TN TP FN

Table 3: Examples of ASR performance evaluation

5 Performance Evaluation on the Dataset

5.1 General Overview

Despite the promising features of WhisperX and Wav2Vec2 models, our initial tests revealed suboptimal performance on our specific dataset, as shown in Figure 6. This underperformance was particularly pronounced for pseudo-words across all models, and for the VOX model across all list types. The latter may stem from the limited scope of the VOX model's training data, which primarily consisted of European Parliament event recordings (VoxPopuli dataset), not aligning well with the characteristics of our child speech dataset. In contrast, the Wav2Vec2 XLS variant demonstrated superior performance over the Vox model. This improvement can be attributed to the XLS model's more extensive pre-training and fine-tuning on a larger and more diverse dataset, enhancing its ability to generalize across various speech contexts. Moreover, datasets used to both pretrain and finetune this model include read speech, which could have improved the model's performance on our dataset featuring children performing reading speech evaluation tasks. Interestingly, WhisperX seems to be aware that it is processing a list of items, as it returns transcriptions consisting of words separated by commas, suggesting some level of contextual understanding in its processing. Given that our transcriptions are based on how the items are heard rather than correct orthographic spelling, they lack consistency, with the same item often transcribed differently across instances. As a result, we cannot reliably compute the Word Error Rate (WER) on our dataset. WER is a metric commonly used to evaluate the performance of ASR systems on datasets by measuring the distance between the recognized text and the reference transcript. This inconsistency in our transcriptions prevents us from comparing the performance of the evaluated models with their performance on other datasets, as our WER calculations would not be accurate or meaningful.

Although WhisperX was trained on a substantial amount of data, it struggled with pseudo-words, indicating a limitation in generalizing to novel word structures. It showed a tendency to fit pseudo-words to more common words (e.g., transcribing the pseudo-word ”rof” as ”oeuf”) or not transcribe them at all. Wav2Vec2 XLS, on the other side, performed better in this regard. This can be attributed to Wav2Vec2’s design as an acoustic model, which relies mostly on acoustic features for transcription in the absence of a language model. However, Wav2Vec2 XLS sometimes struggles to distinguish items and can have a tendency to split a single item into two or to merge two successive items into one. We attribute this to its reliance on acoustic features and the variability in child read speech, where speaking rates can be highly variable, sometimes resulting in words being read consecutively without clear pauses, or pauses occurring within words. Table 4 illustrates each model’s weaknesses in handling novel word structures and maintaining accurate word boundaries:

- **WhisperX** attempt to fit pseudo-words into known French words (e.g., ‘De Querelle’ for ‘bucurelle’, “Jupiter” for ‘hupeur’).
- **VOX** struggles with word boundaries (e.g., ‘ninoirttrafone’ for ‘ninoie trefonet’, ‘troufon du lonpur’ for ‘trufondulence hupeur’).
- **XLS** better word separation and closer to the original item, though it still makes errors (e.g., ‘troufon du lance’ for ‘trufondulence’, ‘turlame’ for ‘turlème’).

Reference	bucurelle daveau ninoie trefonet tiège ronjeau meuil trufondulence hupeur mive gralation turlème
Clinician Transcription	brou bu.curelle davo ni.noi tréchfoné tiège ron.jeau mieu tronfondulence uper mèive car gralation turlem lé {cut recording}
WhisperX	De Querelle, Davos, Nînois, Trésifonée, Thièges, Rongeau, Mieux, Tronçon du Lens, Jupiter, Mives, Quelle relation tu relèves ?
VOX	por u urrel davo ninoirttrafone tiegeronjemiur troufon du lonpur milzegagralation turllaenl
XLS	bbeucurel daveau ni noir tréfoné tiège ron jau mieux troufon du lance uper mive gagralation turlame

Table 4: Comparison of the performance of WhisperX, Vox and XLS on a pseudo words list

5.2 Language Model and Hotwords Integration

A significant portion of false negatives was attributed to spelling mistakes. While accounting for all possible spelling variations would be ideal, it’s impractical due to the extensive permutations involved. To address the spelling issue, we evaluated the XLS model with the integration of a Language Model (LM) and Hotwords. Results are illustrated in Figure 7.

XLS Existing LM: As expected, the XLS model's performance with the existing LM did not show significant improvement. This lack of enhancement is likely due to the nature of our data, which comprises lists of unrelated items, and non-words that are probably not represented in the LM. Consequently, the statistical context provided by preceding words is not helpful for predicting the current word. The non-custom LM slightly decreased the number of true positives for pseudo configurations. This decrease are utterances where pseudo words are correctly by the child (such as 'puli') but the influence of the LM results in another existing similar word being outputed ('puis').

Custom LM and Hotwords: In contrast to the existing LM, integrating a custom LM that included all items from the list led to a significant performance boost. Moreover, giving as hotwords to the XLS model (without a LM) the list of items the child is tasked to read resulted in slightly better performance. For both cases, the integration resulted in around 20% increase in accuracy across all configurations (cf Figure 7). This increase in performance is mostly due to an increase in the number of true positives. This means that there are more words that are said correctly that will be classified as correct by the ASR. This can mostly be attributed to occurrences where the ASR returned a different spelling of the item from the one we expected (e.g. 'passé' instead of 'passer') that are now corrected. However, these integrations also led to a notable increase in false positives, suggesting a risk of overfitting. The rate of false positives with both the custom LM and the hotwords aproach exceeded our threshold of 5% for all configurations, demonstrating an increased tendency to misclassify mispronunciations as correct words.

Reference	bucurelle daveau ninoie trefonet tiège ronjeau meuil trufondulence hupeur mive gralation turlème
Clinician Transcription	brou bu.curelle davo ni.noi tréchfoné tiège ron.jeau mieu tronfondulence uper mèive car gralation turlem lé {cut recording}
XLS no LM	bbeucurel daveau ni noir tréfoné tiège ron jau mieux troufon du lance uper mive gagralation turlame
XLS + LM	deux cures daveau ni noir trépané tiège rongea mieux trou fond du lanceur mive gagralation turlan
XLS + Custom LM	rof rari tuit cutice bumon bante perné loi bonbonfange je tetraime
XLS + Hotwords	bucurelle daveau ninoie trefonet tiège ronjeau meu trufondu-lance hupeur mive gralation turlème

Table 5: Influence of a LM and Hotwords for the XLS model on a pseudo words list

Ambiguity in Text Based Outputs Looking more closely at the items generating a significant number of false positives, we realized that evaluating mispronunciations from text outputs poses substantial challenges in accurately classifying the model's output illustrated by the following examples:

- When considering the word 'orchestre' (/ɔʁkɛstʁ/), there might be instances where a child pronounced 'orchestre' with a /ʃ/ sound (resulting in /ɔʁʃɛstʁ/) instead of the correct /k/ sound. Furthermore, other words in the training datasets where the digraph 'ch' is pronounced as /ʃ/ (e.g., 'chat' /ʃa/) may have influenced the model's learning, making it difficult to determine whether the model is transcribing based on correct pronunciation or learned spelling patterns.

- A similar issue arises with words like 'aout': The correct pronunciation is /ut/, but a possible mispronunciation is /aut/. Classifying pronunciation based on this text output doesn't allow us to definitively determine which pronunciation the model is recognizing when it outputs 'aout'.

These examples highlight the ambiguity in text-based representation of pronunciations underscoring the limitations of relying solely on textual output for assessing the model's ability to distinguish between correct pronunciations and mispronunciations.

5.3 Phonetic Transcription Approach

An alternative strategy we explored involved classifying the output using International Phonetic Alphabet (IPA) phonetic transcriptions. In this approach, IPA-encoded transcriptions were matched against the IPA-encoded items of the reference text. Despite the Wav2Vec2 Phonemizer model being pre-trained on significantly less data compared to other ASRs (approximately five times less than the VOX model, 22 times less than XLS, and 30 times less than Whisper) and being trained on a single dataset while both WhisperX and XLS were trained on various data, it still demonstrated promising performance.

When compared to the VOX model, which is trained on a larger subset of the same dataset, the Phonemizer showed notable advantages. Its performance was competitive with both Whisper and XLS across simple and complex configurations, and it outperformed these models in detecting pseudo-words. This suggests that explicit phonetic training can be highly effective for detecting mispronunciations and managing unknown words.

5.3.1 Challenges in Text-to-Phonetic Conversion of the Transcriptions

Although promising, the model is not sufficiently performant to be used off-the-shelf, and finetuning it on our data poses a real challenge as the nature of our transcriptions prevents their conversion into phonetic encoding.

Encoding Based on Known Mappings Models relying on pre-defined mappings often struggle with non-words, misspelled, or accented characters, leading to erroneous encodings. Given the nature of our transcription, which features pseudo-words and spellings reflecting how words are heard rather than their correct orthographic forms, such models are unreliable for phonetic encoding. This can result in errors where a child's mispronunciation is incorrectly encoded as the correct pronunciation due to pre-existing mappings of the clinician transciption, as illustrated in Tables 6

Pronunciation	Phonetic Encoding	Clinician Transcription	Convertor's Output
Correct	/ɔʁ**k**ɛstʁ/	or**k**estre	/ɔʁ**k**ɛstʁ/
Incorrect	/ɔʁ**ʃ**ɛstʁ/	or**ch**estre	/ɔʁ**k**ɛstʁ/
Pronunciation	**Phonetic Encoding**	**Clinician Transcription**	**Convertor's Output**
Correct	/apaʁi**s**jɔ /	appari**ss**ion	/apaʁi**s**jɔ /
Incorrect	/apaʁi**t**jɔ /	appari**t**ion	/apaʁi**s**jɔ /

Table 6: Mismatches between Text-to-Ipa's output and correct phonetical encoding of the utterance: "orchestre" and "apparition": examples

Literal Sound Translation Issues: Models that translate each sound literally can lead to misrepresentations, particularly in languages like French with many silent letters. The inconsistent spelling in our transcriptions exacerbates these issues, resulting in potential incorrect encodings. For example

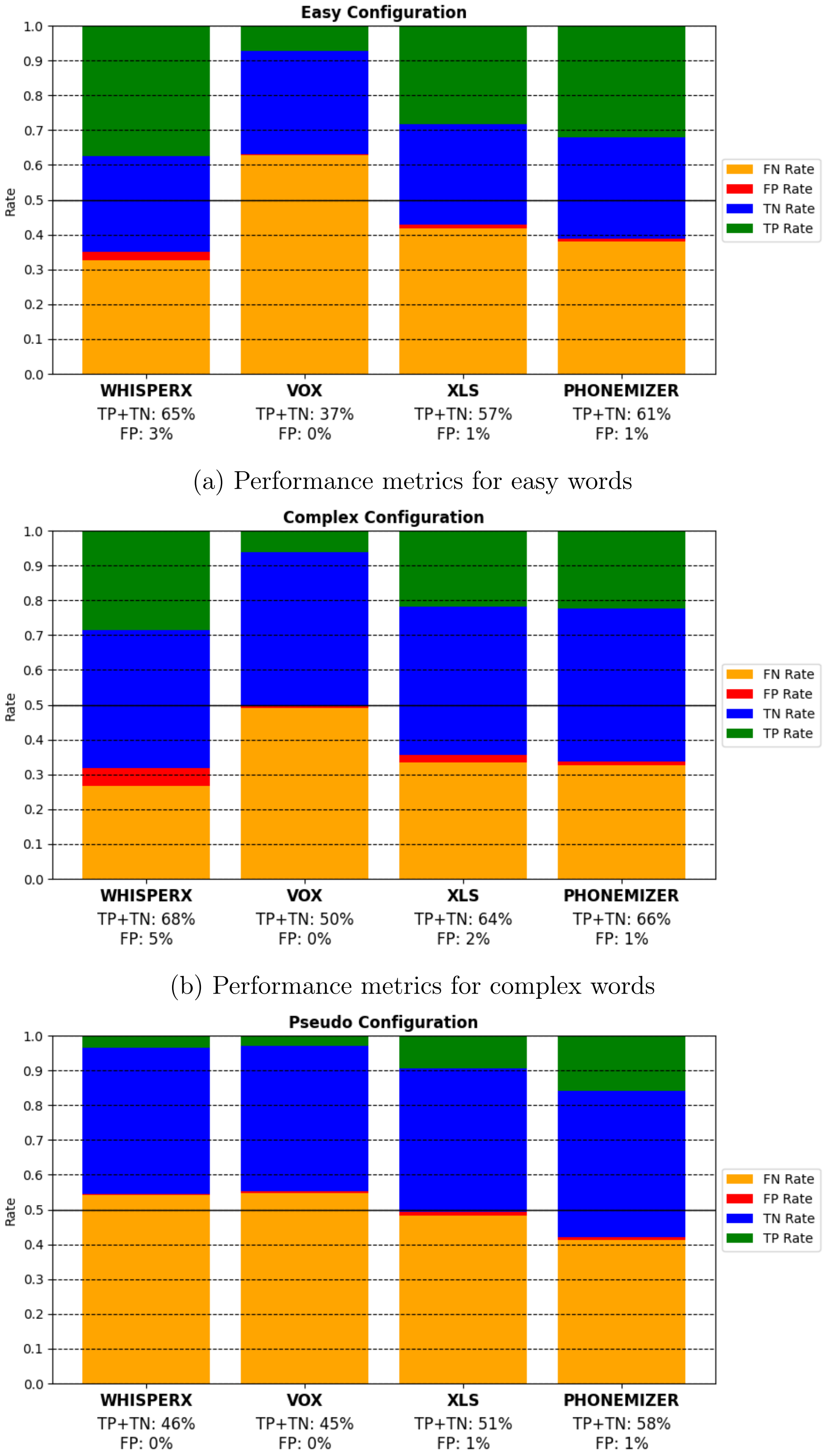
- The pseudo-word 'lisoie' pronounced /lizwa/ might be incorrectly encoded with an /s/ instead of a /z/.
- The silent 'f' in the french word 'cerf', pronounced /sɛʁ/, would be incorrectly marked in its phonetic translation resulting in /sɛʁf/.

These limitations, due to the way the transcription are encoded, rendered it unfeasible to fine-tune a Wav2Vec2 phoneme model on our data, as the underlying phonetic encodings would be unreliable.

5.4 Summary of the Challenges Encountered

The suboptimal performance of off-the-shelf models underscores the challenging nature of our task, which combines child speech recognition with the added complexities of non-words and unrelated words within a list. The effects of incorporating an LM and hotwords, while increasing accuracy, led to overfitting and an increase in false positives, with rates exceeding our threshold of 5%. This outcome is particularly problematic in our context, as the false positives represent mispronunciations that go undetected by the model. Consequently, these errors will be scored as correct pronunciations and won't be reviewed by a professional, compromising the accuracy of the speech evaluation process.

Moreover, our dataset consists of children from the South of France, who speak with a very specific regional accent. For instance, 'rose' might be pronounced with a more open vowel sound in the South. Due to this regional specificity, fine-tuning a model on our dataset might not yield good performance on datasets with different accents. Additionally, because our data is tailored to specific lists of (pseudo) words, any changes in the corpus could necessitate redoing the fine-tuning. Fine-tuning models also requires knowledge, time and resources, which may not always be available to professionals, making it an impractical solution. These findings highlight the need for an alternative and more flexible approach to evaluating mispronunciations in child speech recognition tasks, allowing for easy adaptation to different datasets and accents without extensive retraining.

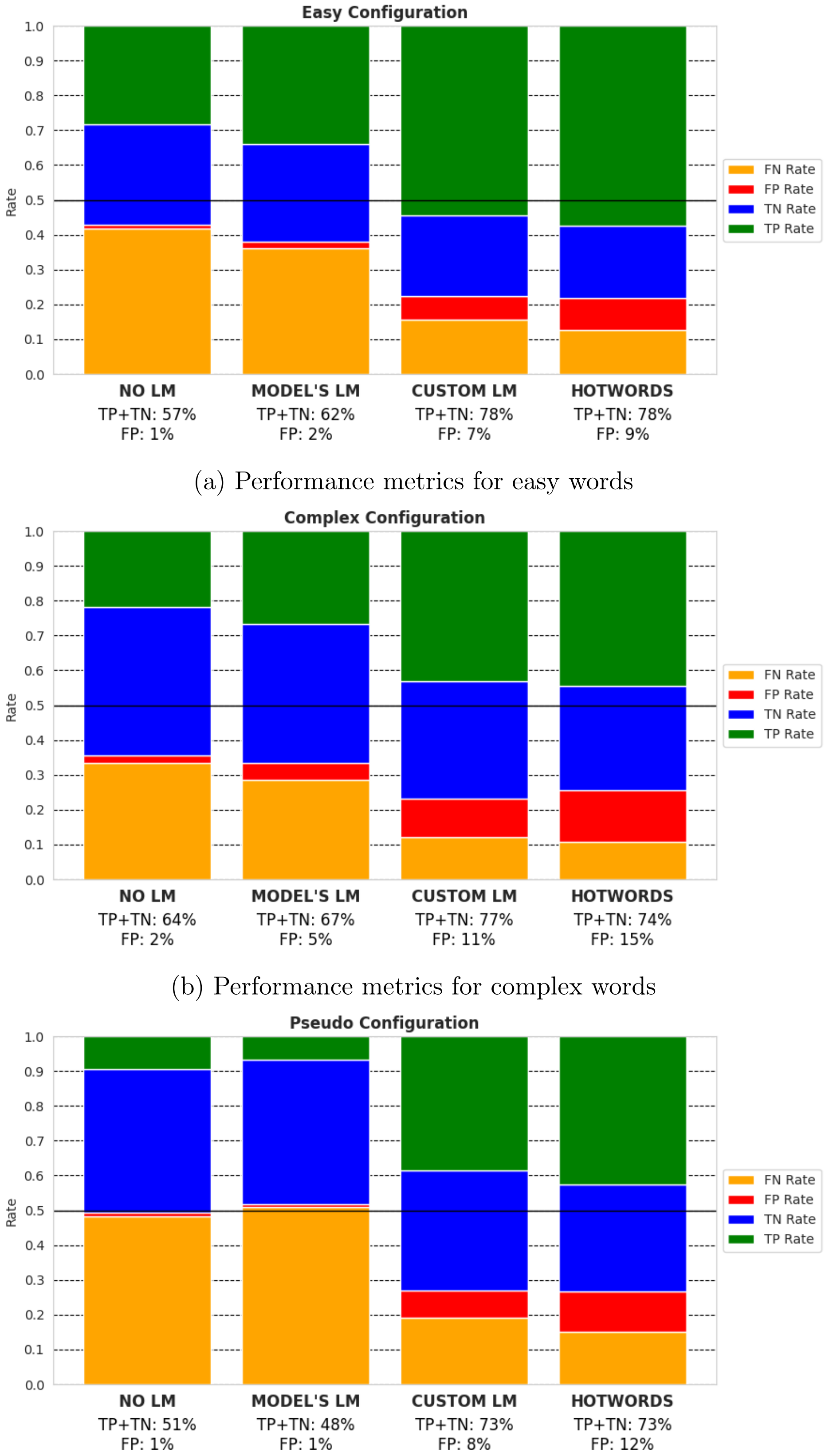


(a) Performance metrics for easy words

(b) Performance metrics for complex words

(c) Performance metrics for pseudo words

Figure 6: Performance comparison of different models across lists types



(a) Performance metrics for easy words

(b) Performance metrics for complex words

(c) Performance metrics for pseudo words

Figure 7: Effect of a language model and hotwords on the Wav2Vec2 XLS model

6 ASR for Mispronunciation Detection: Pipeline Overview

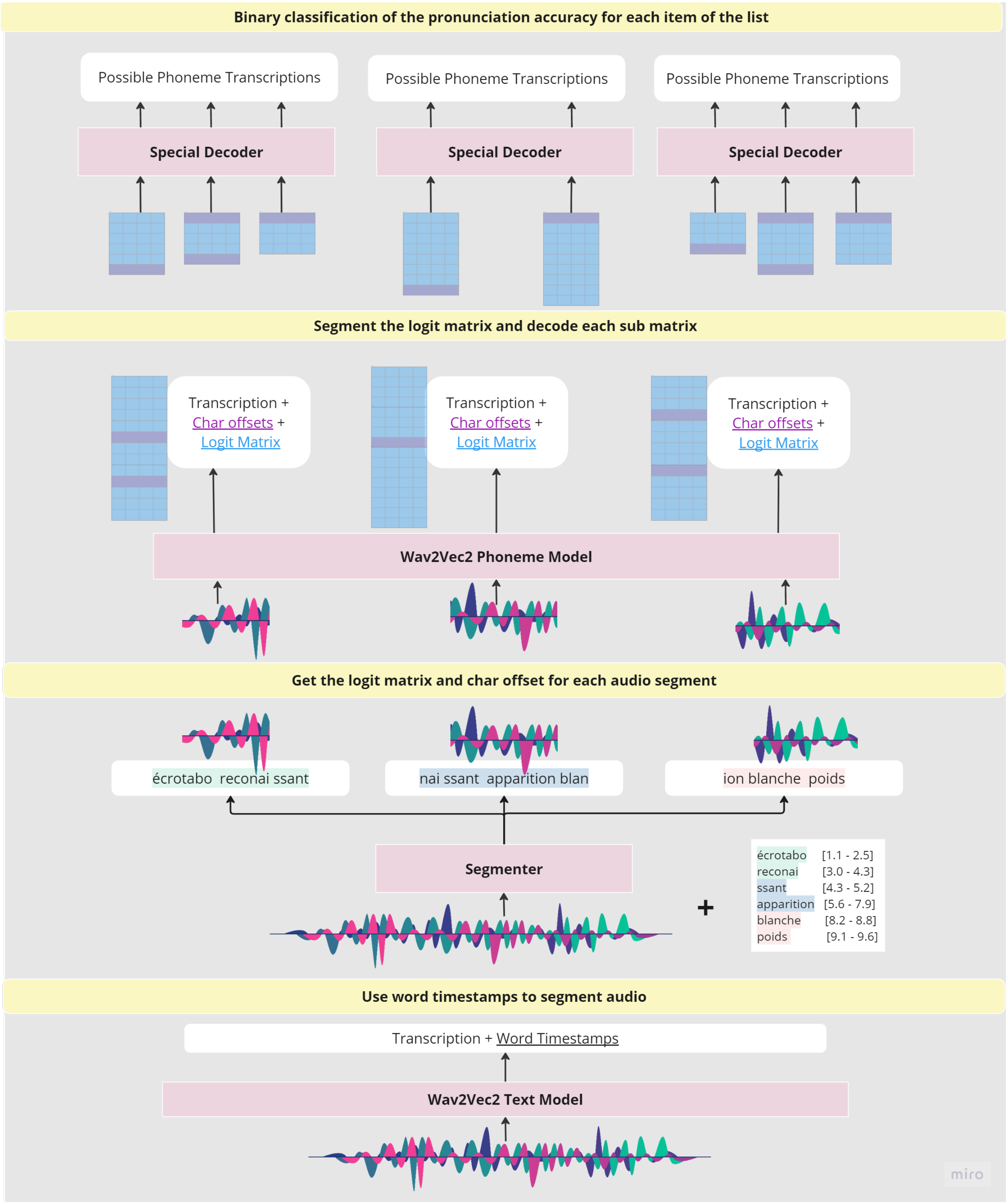
While we cannot train the Wav2Vec2 Phonemizer model due to the aforementioned challenges, initial observations revealed that it performed better when processing shorter segments of the audio rather than the whole file. This insight led to the development of a multi-stage pipeline, illustrated in Figure 8:

1. **Hotword-Enhanced Wav2Vec2 Model:** The audio input is first processed by a Wav2Vec2 Text Model finetuned on adult French data. This model, which we call the ASR segmenter, is enhanced with the list of (pseudo) words as hotwords, and returns a transcription along with the start and end timestamps for each identified item. As demonstrated in section 5, the use of hotwords significantly enhances the model's ability to correctly identify items from the lists, including pseudo-words. This strategy minimizes cases where items are incorrectly split or merged when processing our unique dataset. The enhanced model is better equipped to handle the nature of our data and the variations in children's speech, including differences in rhythm, phoneme duration, and pause insertion patterns. We opted for hotwords enhancement rather than LM integration because hotwords can be easily provided as parameters during inference, whereas LM generation and integration into the model requires technical expertise and is less practical for professionals to implement. Moreover, our experiments showed that hotwords achieved better accuracy, even if it leads to overfitting, which is not a concern at this stage as we only use the timestamps, not the transcriptions. By prioritizing accurate item timestamps over avoiding overregularization, we ensure more reliable segmentation for subsequent processing stages.

2. **Audio Segmentation:** Using the word timestamps obtained in step 1, the audio is segmented into shorter chunks. This segmentation allows for more accurate phoneme recognition in the subsequent steps. More details on the segmentation process and its impact will be discussed in subsection 7.

3. **Phoneme-Level Processing:** Each audio segment is then processed individually by a Wav2Vec2 Phoneme Model fine-tuned on an adult French dataset. This model operates without the use of a LM or hotwords to avoid overfitting, a concern at this stage. The model outputs a logit matrix (representing the confidence scores for each possible character at each timeframe of the audio input, cf Section 8), and character timestamps for each segment, allowing for precise phoneme recognition.

4. **Binary Classification via Decoding:** The logit matrix for each segment is further divided based on the character offsets. A specialized decoder processes each sub-matrix to generate potential phoneme transcriptions for each item. These generated transcriptions are then compared to the expected pronunciations to score the accuracy of pronunciation of each item (correct/incorrect). We will explore this decoding process's technical details and effectiveness in later sections.


Binary classification of the pronunciation accuracy for each item of the list
Possible Phoneme Transcriptions
Possible Phoneme Transcriptions
Possible Phoneme Transcriptions
Special Decoder
Special Decoder
Special Decoder
Segment the logit matrix and decode each sub matrix
Transcription + Char offsets + Logit Matrix
Transcription + Char offsets + Logit Matrix
Transcription + Char offsets + Logit Matrix
Wav2Vec2 Phoneme Model
Get the logit matrix and char offset for each audio segment
écrotabo reconai ssant
nai ssant apparition blan
ion blanche poids
Segmenter
+
Use word timestamps to segment audio
Transcription + Word Timestamps
Wav2Vec2 Text Model
miro

écrotabo	[1.1 - 2.5]
reconai	[3.0 - 4.3]
ssant	[4.3 - 5.2]
apparition	[5.6 - 7.9]
blanche	[8.2 - 8.8]
poids	[9.1 - 9.6]

Figure 8: Pipeline for using ASR as a tool to detect mispronunciation

This pipeline, designed to integrate ASR as a first-pass tool for detecting mispronunciations in clinicians' speech evaluation workflows, effectively combines the strengths of both word-level and phoneme-level models. The initial segmentation based on word-level timestamps establishes a reliable framework for the subsequent phoneme-level processing.

We utilized Wav2Vec2 models fine-tuned on adult speech data because they are more readily accessible, and easier to finetune if not available due to the abundance of adult speech datasets. In contrast, child speech datasets, particularly in low-resource languages, are scarce to inexistant. This approach balances practicality and accuracy, allowing us to leverage available resources while addressing the unique challenges posed by our dataset.

7 Optimal Audio Segmentation

When transcribing entire audio files phonetically, we encountered significant issues with item boundary delimitation, phoneme omissions, and errors. We were unable to compute the Phoneme Error Rate (PER), a metric that quantifies the discrepancy between recognized and reference phoneme sequences, due to the absence of phonetically encoded transcriptions. This limitation hindered our ability to directly compare the model's performance on our datasets with its performance on others. However, we hypothesize that some of the difficulties may be attributed to the distinct variations present in children's speech patterns. We noticed that in our dataset, speaking rates can be highly variable, sometimes resulting in words being read consecutively without clear pauses, or pauses occurring within words. Since the Wav2Vec2 phonemizer operates without the help of a language model or hotwords, it lacks linguistic information to guide word boundary decisions, making it more susceptible to bad item delimitations.

In this section, we will discuss our final strategy to overcome these issues. The results for our initial item-level segmentation approach, showing the importance of a good padding duration, are detailed in Appendix A, while the results for our final approach are shown in Figure 10.

7.1 Addressing Model Training Mismatch

Taking a closer look at the phonemizer model finetuning data, we observed that it was trained on audio samples averaging 5-6 seconds in length. In contrast, our dataset contained full audio samples averaging 20-30 seconds, with individual word pronunciations being very brief and lasting from a few milliseconds to a few seconds. To address this mismatch, we initially segmented the audio into 5 seconds chunks (ensuring no item was split between segments) with 1 second of overlap between segments. However, this adjustment decreased performance. Upon closer inspection, we noticed that some audio segments contained almost no speech due to children speaking slowly or making pauses, resulting in 5s audio segments with minimal actual speech content. Some children will read the item very fast and have 5 seconds of audio full of speech content, while others will read very slowly or make pauses, resulting in 5s of audio with very little speech content. This observation highlights the unique challenges posed by children's read speech patterns as they show greater variability in the rhythm of speech which can impact the performance of models.

7.1.1 Final Approach

To optimize our approach, we need to consider not just the duration of the audio segments, but also the actual speech content within those segments. This insight led us to our final solution illustrated in Figure 9 and consisting of:

1. **Segmenting audio based on actual speech content:**
   Instead of using fixed durations like 5 seconds for each audio segment, we adjust the segments to a variable length based on actual speech content using the timestamps returned by the ASR Segmenter, targeting X seconds of actual speech.

2. **Adding 1 second of padding before and after each segment:**

   (a) Boundary Effect Mitigation: As discussed in Appendix A, adding of 1 second at segment boundaries yields better results, especially at the segment boundaries.

   (b) Compensation for the ASR Segmentation Errors: Children's speech often includes irregular pacing, such as extended pauses or rapid transitions. While hotwords help mitigate these issues, segmentation errors can still occur. Padding addresses cases where items at segment boundaries are split or misaligned, increasing the chance of capturing the entire item in at least one segment.

The example in Figure 9 illustrate the final approach for $X = 3$. Here each resulting segment contains at least 3 seconds of actual speech. The padding at the segment's border helps overcome issues caused by the ASR Segmenter's inaccurate item delimitation, as illustrated by the word 'reconaissant' being fully included in the first segment. Additionally, the approach effectively handles the long pause between the words 'ssant' and 'apparition' by focusing on actual speech content, thereby avoiding segments with little to no speech.

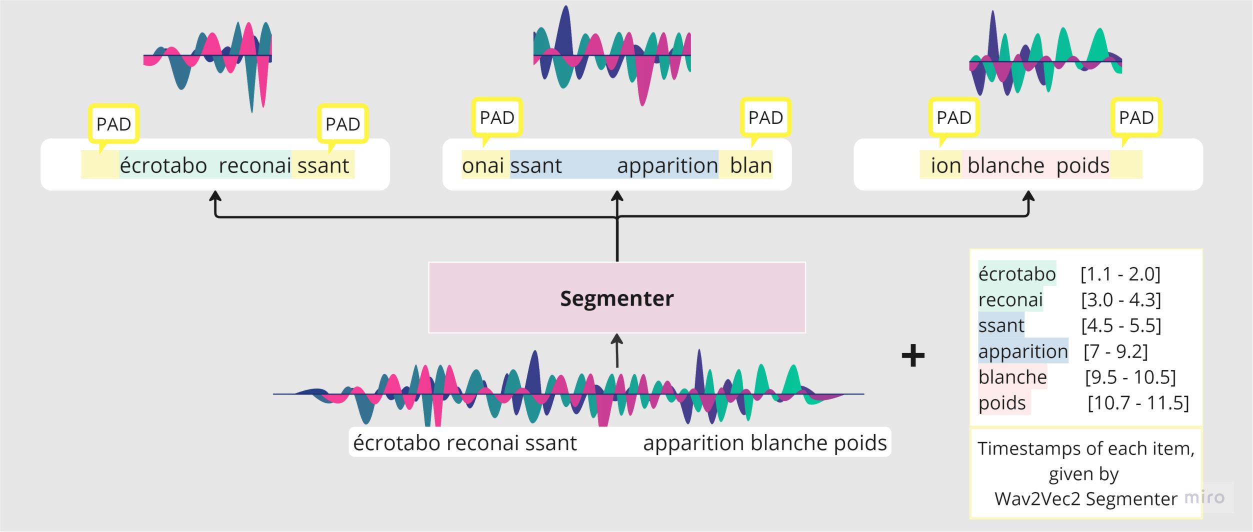


Figure 9: The final segmentation approach with $X = 3$.

Performance Evaluation: We tested values of X in {1,2,3,4,5,7,10} seconds and found:

- All values of X outperformed whole-file processing for each configuration. Performance gains were generally consistent across different values of X, but larger values of X led to a slight decrease in performance.
- The optimal segment duration was found to be X=4 seconds, which resulted in a 3.3% increase for the easy complex configuration, 3.6% and and 1.5% for the pseudo configuration compared to whole-file processing. The smaller improvement for pseudo configurations highlights the persistent challenge in processing these particular items, stemming partly from poorer item segmentation.

The choice of X reflects a balance between capturing sufficient context and avoiding the drawbacks of excessively long segments. Values of X higher than 4 seconds did not improve performance, and setting $X = 10$ seconds even led to a decrease in performance. This indicates that, for our data, while segments that are too short do not yield good performance, too much context may not provide additional benefits and could potentially dilute the focus on individual items. This segmentation strategy offers a robust approach to improving transcription accuracy by handling boundary issues and providing the optimal amount of context in each resulting segment, improving overall phoneme recognition.

8 Wav2Vec2 Standard Decoding Process

The Wav2Vec2 model generates a logit matrix as its output, which can be characterized by several key features. The matrix has dimensions of `[Number of timeframes × Number of characters in the vocabulary]`, where each timeframe represents 20 milliseconds of the audio input. The vocabulary consists of all possible output tokens of the phoneme recognizer, totalling 59 distinct characters for the Wav2Vec2 Phonemizer model we considered (see Figure 11). For instance, an audio input of 1 second would produce approximately 50 timeframes, resulting in a logit matrix with dimensions of `50 × 59`.

Each cell in the matrix contains a logit value, which is a raw, unbounded score reflecting the model's confidence that a particular character is present at a given timeframe. Higher logit values indicate greater confidence. Figure 14 in Appendix B illustrates a simplified example of a logit matrix

8.1 Decoder Process

In the standard decoding process of Wav2Vec2, an argmax operation is applied to the logit matrix to extract the most probable transcription. For each timeframe, the decoder identifies the character corresponding to the highest logit value and selects it as the most likely output for that moment in time. This step is repeated across all timeframes, producing a sequence of phonemes that represents the model's best guess at the speech content. After this initial selection, the sequence undergoes further processing to remove consecutive repetitions of the same character and to eliminate any pad tokens. Finally, the transcription is returned, along with the character timestamps.

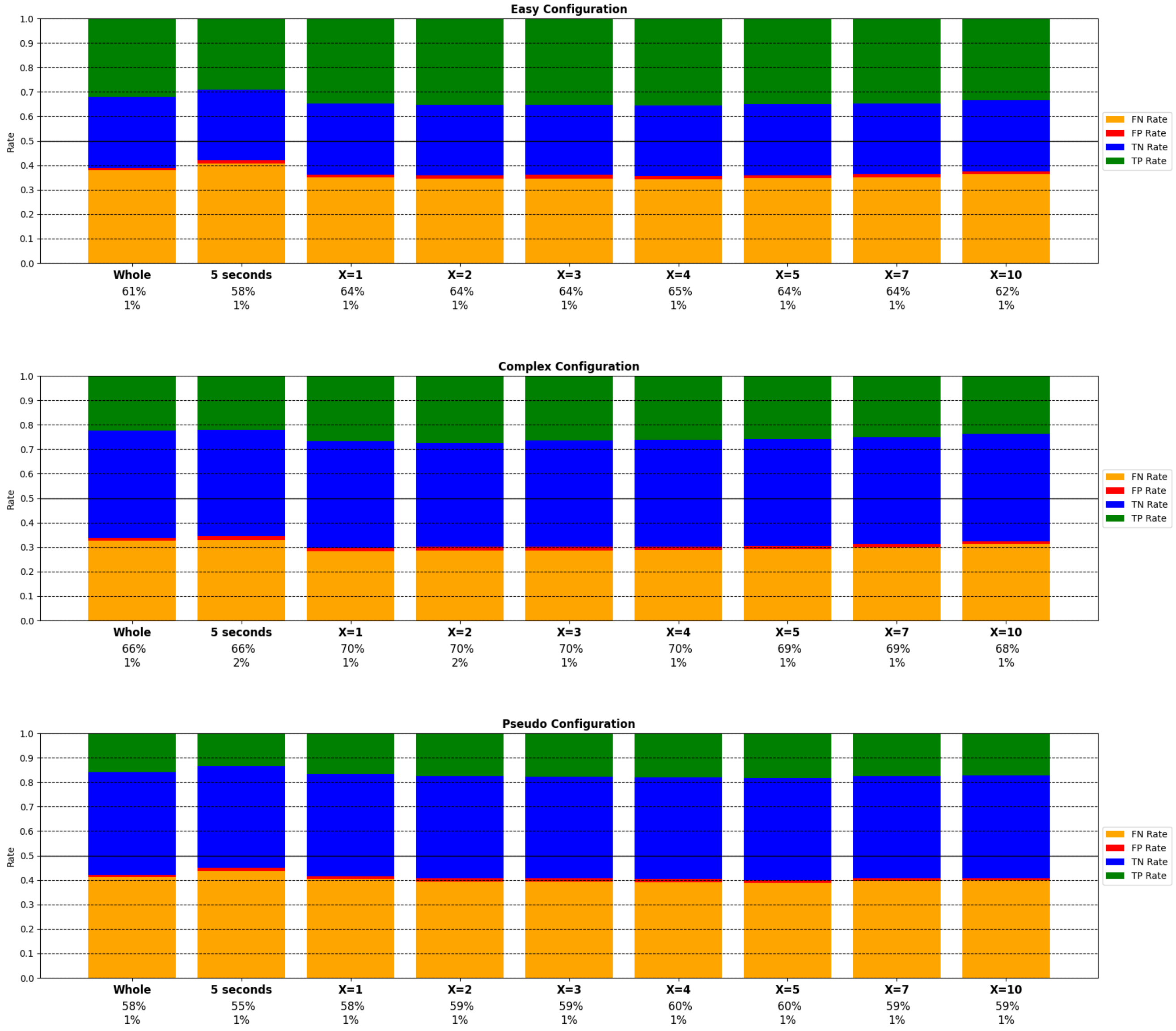
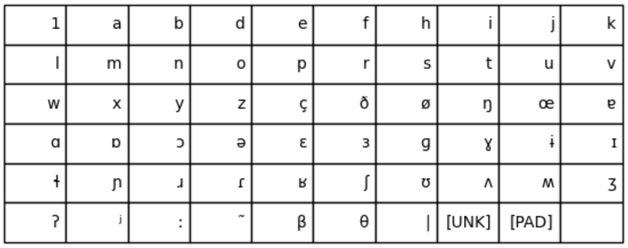


Figure 10: Comparison of model performance across all configurations for different segmentation approaches. Strategies include whole-file processing, 5-second segments, and our final approach with varying X values. Percentages shown are TP+TN (top) and FP (bottom).



1	a	b	d	e	f	h	i	j	k
l	m	n	o	p	r	s	t	u	v
w	x	y	z	ç	ð	ø	ŋ	œ	ɐ
ɑ	ɒ	ɔ	ə	ɛ	ɜ	ɡ	ɣ	ɨ	ɪ
ɫ	ɲ	ɹ	ɾ	ʁ	ʃ	ʊ	ʌ	ʍ	ʒ
ʔ	ʲ	ː	˜	β	θ	|	[UNK]	[PAD]	

Figure 11: The 59 tokens used by the ASR Phonemizer

8.2 Limitations

While this approach is computationally efficient and generally effective, it has significant limitations when applied to our dataset, as illustrated in the example in Appendix B.1.1:

- Single Character Consideration: The decoder only considers the single most probable character at each timeframe, potentially discarding valuable information from close candidates. This can be particularly problematic in cases where multiple phonemes have very close logit values.
- Sensitivity to Pronunciation Variations: Child speech often involves more variable pronunciation patterns. The rigid selection of the most probable character may not adequately capture these variations, leading to potential transcription errors.
- Vulnerability to Noise: In the presence of background noise or microphone buzzing, the highest probability character may not always correspond to the correct phoneme.

9 Enhanced Decoding Process

In this section, we delve into the evolution of our decoding approaches, examining the methodologies, limitations and improvements implemented across different decoder versions. The comparative performance of these decoders are illustrated in Figure 12.

9.1 Decoder 1: Considering Multiple Probable Outputs

To address the standard decoder's limitations and improve the accuracy of our phoneme recognition, we propose a more sophisticated decoding strategy which involves considering multiple probable phonemes at each timeframe. This decoder operates on the logit matrix and the character offsets outputed by the Wav2Vec2 phonemizer and proceeds as follows:

1. **Initial Segmentation**:

   Input: Full segment logit matrix and predicted space character offsets outputed by Wav2Vec2 Phonemizer.

   Output: Set of item-level submatrices, each corresponding to a predicted (pseudo)word.

   The process begins by segmenting the logit matrix based on predicted space offsets. This segmentation aims to isolate individual items. Importantly, the space characters are included in the subsequent submatrices to account for potential errors in the initial prediction. This item-level approach allows for more efficient handling of different phoneme possibilities compared to processing the entire logit matrix at once.

2. **Multiple Candidate Selection (for each submatrix)**:

   Input: Item-level submatrix

   Output: List of candidate phonemes for each timeframe.

   The algorithm selects at most k phonemes at each timeframe whose logit values are within a threshold t of the top candidate. This approach allows for the consideration of closely competing phonemes.

3. **Combinatorial Prediction Generation (for each submatrix)**:

   Input: Lists of candidate phonemes for each timeframe.

   Output: List of plausible phoneme sequences for the submatrix's item.

   Using the expanded candidate pool, the algorithm generates all plausible phoneme sequences for each submatrix through combinatorial generation. Each sequence in this list is then processed by removing consecutive repetitions of the same character and any pad tokens, resulting in a list of possible phonetical encodings of the considered item. The number of predictions generated per item is capped at a maximum value M.

4. **Target Matching**:

   Input: List of the possible phonetical encodings of the item considered and list of the phonetical encodings of each target of the reference text.

   Output: Accuracy score

   For each target item not evaluated yet, we compare its encoding with the list of possible phonetical encodings of the current item considered. If any of the generated sequences match with the target, we score that target at 1 indicating successful pronunciation.

9.1.1 A Tunable Process

This decoding process enhances our ability to accurately recognize phonemes in child speech by accounting for the higher variability and potential ambiguities in pronunciation, and the noise present in the audio. The process's flexibility is achieved through several tunable parameters, with the threshold t being particularly crucial:

- **Threshold t**: The raw score threshold for including additional phoneme candidates. Its significance lies in its ability to adjust the balance between false positives and false negatives, making it a key feature of our pipeline as we can optimize the system for different clinical and research applications.

  By adjusting t, we can address both scenarios discussed in Section 4.1:

  - Higher t values increase sensitivity, capturing more potential phoneme candidates. This is beneficial when the ASR is used for real-time task assessments, where minimizing FNs is crucial to avoid premature task termination.
  - Lower t values increase specificity, considering only the closer phoneme candidates. This is advantageous when the ASR is used for post-hoc analysis, where minimizing FPs is essential to ensure no errors are overlooked.

  As stated earlier, we focus this study on assessing performance with respect to the post-hoc analysis scenario. In line with this focus, we conducted testing with various threshold values. We determined $t = 2$ to be optimal, achieving the best performance while maintaining our FP rate below the critical 5% threshold. This setting ensures that only very close phoneme candidates are considered, reducing noise in the decoding process while still allowing for some flexibility to account for pronunciation variations in child speech.

It's important to note that while we've optimized for this specific scenario, our system's flexibility allows for easy adaptation to other use cases, such as real-time task difficulty assessment, through adjustment of the t value.

- **Top-k selection** k: The number of top phoneme candidates considered at each timeframe.

  We chose $k = 59$, which is equal to the number of characters in our vocabulary. This means considering all phonemes at each timeframe. This choice was made because, with our conservative threshold t, only a few characters at each timeframe typically satisfy the condition of having their logit value under the threshold. However, this approach allows for computational feasability if we were to use a more liberal threshold.

  For instance, if we set a threshold $t = 5$, we might find 15 phonemes respecting the condition at a single timeframe. In such cases, setting a smaller k would be beneficial to limit computational complexity. By keeping k equal to our vocabulary size, we ensure we're not arbitrarily limiting our phoneme options while our conservative t naturally restricts the number of candidates.

- **Maximum predictions** M: The maximum number of predictions generated per item.

  We set $M = 15$ to balance between computational feasibility and a comprehensive exploration of phoneme combinations. This cap ensures that the process remains computationally feasible as for some items, there are multiple timeframes with different possible phonemes selected, which can lead to a combinatorial explosion of possibilities.

  For instance, if there are 5 possibilities for timestamp 1, 2 for timestamp 8, and 3 for timestamp 10, we would have $5 * 2 * 3 = 30$ possibilities. Without a cap, we would have 30 number of combinations to process. Setting $M = 15$ allows us to explore a reasonable number of the most likely combinations without overwhelming computational resources.

These parameter choices allow our decoder to be both flexible and computationally manageable. In our case, the conservative threshold t ensures precision in phoneme selection, while the high k value and moderate M value allow for exploration of multiple plausible phoneme sequences. Specifically, the tunability of the threshold t is a key advantage in the context of child speech evaluation task, as it allows us to adjust how conservative or liberal we want to be in phoneme selection. For our current task, which requires high precision, we set a relatively conservative $t = 2$. However, for other tasks where recall might be more critical, we could set a higher t value to include more potential phoneme candidates. This adaptability makes our decoder suitable for a wide range of speech evaluation scenarios.

The example in Appendix B illustrate the effectiveness of Decoder 1 in overcoming the limitation of to the standard Wav2Vec2 decoder. Using this decoder on our enhanced pipeline provides more robustness when handling the variability of noisy child speech as it allowed for very probable outputs to be considered. This resulted in an average accuracy improvement of 7.5% over the standard decoder, as shown in Figure 12. Specifically, the accuracy increased as follows:

- **Easy configuration:** From 63.7% to 72.5% (8.8% increase)
- **Complex configuration:** From 69.7% to 75.1% (5.4% increase)
- **Pseudo configuration:** From 59.8% to 67% (7.1% increase)

However, this improvement in accuracy was accompanied by a rise in false positive rates, increasing overall from 1.5% to 3.3%, though remaining well under our threshold of 5%.

9.1.2 Decoder 1 Limitations

The first decoder, while more performant than its predecessor, struggles with inaccuracies in space character timestamps outputted by Wav2Vec2. Whether due to missing or extra spaces, Decoder 1 will fail to accurately assess pronunciation as illustrated by the examples featured in Appendix C.1.1 & C.2.1. This approach of treating each predicted submatrix in isolation and requiring exact matches between probable outputs and the targets leads to missed identifications when word boundaries are incorrect. This highlights Decoder 1's lack of robustness in handling the variability of item boundaries, a challenge for Wav2Vec2 Phonemizer on our dataset as discussed in Appendix A.

9.2 Decoder 2: Improved Handling of Item Boundaries

To address these limitations and improve the accuracy of the pipeline, we propose the following improvement over Decoder 1:

1. **Initial Segmentation:** Same as Decoder 1
2. **Multiple Candidate Selection:** Same as Decoder 1
3. **Combinatorial Prediction Generation:** Same as Decoder 1
4. **Concatenation of Adjacent Predictions:**

   Input: List au plausible transcription for each item.

   Output: List of concatenated transcriptions using a sliding window of size 3

   Instead of treating each predicted subsegment in isolation, Decoder 2 considers combinations of adjacent predictions. For each subsegment, it concatenates all elements from:

   - The prediction list from the previous subsegment
   - The prediction list from the current subsegment
   - The prediction list from the next subsegment

   This creates a sliding window of size 3 of concatenated predictions, allowing for the detection of words that may have been split across segment boundaries.

5. **Flexible Matching:**

   Input: List of concatenated transcriptions using a sliding window of size 3

   Output: Accuracy score

   The decoder checks if any of the targets are **contained** within these concatenated predictions, rather than requiring an exact match. This allows for the correct identification of items even when they are partially split or merged with adjacent items.

Examples in Appendix C illustrate the effectiveness of Decoder 2 in overcoming the limitation of Decoder 1. Using this decoder on our enhanced pipeline resulted in a more robust handling of the item segmentation errors, a prevalent challenge of the model on our dataset especially with pseudo words. This resulted in an average accuracy improvement of 5.2% for Decoder 2 over Decoder 1, as shown in Figure 12. Specifically, the accuracy increased as follows:

- **Easy configuration:** From 72.5% to 77.7% (5.2% increase)
- **Complex configuration:** From 75.1% to 78.8% (3.5% increase)
- **Pseudo configuration:** From 67% to 74.4% (7.4% increase)

However, this improvement in accuracy was accompanied by a significant rise in false positive rates, which doubled, surpassing our 5% threshold across all configurations. Notably, the substantial increase in accuracy for the pseudo configuration is attributed to the Wav2Vec2 Phonemizer's particular difficulty with correctly delimiting pseudo-words. Our decoder's robustness against these segmentation errors has thus led to improved performance for these challenging items.

9.2.1 Limitations of Decoder 2

While Decoder 2 showed significant improvements over Decoder 1, particularly in handling Wav2Vec2's inconsistent space character predictions, it introduced a new challenge: an increase in false positive rates. This increase is particularly important for items containing silent letters at their boundaries. The problem stems from Decoder 2's containment-based matching strategy, which struggles to differentiate between correct pronunciations and mispronunciations involving the addition of phonemes before or after a word. This limitation is especially problematic in French, a language featuring many words with silent letters at the beginning or end. For example:

- In the word 'cerf', the final 'f' is silent. Decoder 2 will incorrectly accept a pronunciation that includes the /f/ sound as it is checking if the target /sɛʁ/ is contained in a concatenated list featuring /...sɛʁf../ .
- Similarly, for 'aout', the first 'a' is not pronounced. Decoder 2 will fail to flag a mispronunciation that vocalizes this /a/.

This shortcoming is particularly challenging as it allows mispronunciations to go undetected, introducing bias the the overall speech evaluation process.

9.3 Decoder 3: Improved Error Detection at Boundaries

Decoder 3 builds upon Decoder 2's foundation, addressing its limitations by introducing two error lists for each target, and an error checking mechanism step in the decoder's process. This approach allows a better handling of insertions and subtle mispronunciations at item boundaries that were previously undetected and operates as follows:

1. **Initial Segmentation:** Same as Decoder 1 and 2
2. **Multiple Candidate Selection:** Same as Decoder 1 and 2
3. **Combinatorial Prediction Generation:** Same as Decoder 1 and 2
4. **Concatenation of Adjacent Predictions:** Same as Decoder 2
5. **Error Check Before Containment Check**:

   Input: List of concatenated transcriptions using a sliding window of size 3

   Before checking if a target is contained within the concatenated predictions list of [itemA, itemB, itemC], we first verify if any element from the target's error lists appears in it. Specifically, if an error target is found we set the accuracy score for that target to 0, indicating a mispronunciation. This step halts further checks for this specific concatenation to avoid the risk of false positives
6. **Flexible Matching**: Same as Decoder 2, only if no error is flagged

9.3.1 Error List Creation

For each target item, we generate two error lists that capture potential mispronunciations:

Error List End: [item]+[phoneme] for each phoneme that might be incorrectly added to the end of the item

Error List Beginning: [phoneme]+[item] for each phonemes that might be incorrectly added to the beginning of the item

The error lists need to be generated by clinicians in advance, based solely on the reference texts (the lists of words or pseudo-words to be read) rather than on actual child utterances. The creation of these lists is quick and straightforward, following the guidelines detailed below. The result is a structured data format that serves as input to the decoder. For each target, this format includes a list containing the different possible phonetic encoding, an "Error List End" for potential errors at the end of the word, and an "Error List Beginning" for potential errors at the beginning of the word. This organized structure allows for easy input of the error lists into the decoding system. The guidelines for creating these lists are as follows:

Silent Letters At Boundaries

- For Error List End: We add the corresponding phoneme of any silent letter found at the end of the item. For example, the target 'poids', phonetically encoded as /pwa/, we add /pwad/ to its Error List End.

- For Error List Beginning: We add the corresponding phoneme of any silent letter found at the beginning of the item. For example, the target 'août', phonetically encoded as /ut/, we add /aut/ to its Error List Beginning.

Nasal Sounds In Wav2Vec2 outputted phonetic transcription, the /˜/ character (tilde) is placed after the character it modifies, which can change the sound of a phoneme. To detect mispronounced final phonemes, the decoder automatically adds the /˜/ phoneme to the Error List End (meaning that they do not be considered when encoding the possible errors for a particular dataset).

- Example: For the target item 'beau' (/bɔ/), we add /bɔ̃/ to the error target list. This allows to flag as incorrect cases where a child incorrectly nasalize the word, pronouncing it as 'bon' (/bɔ̃/). The previous decoder would have marked this nasalized pronunciation as correct because /bɔ/ is contained in /bɔ˜/.

Overlapping in Reference Text We also address cases where the phonetical encoding of the current target is contained within another target or in the concatenation of multiple targets. Consider the following scenario:

Reference Text	récit sept beau six
Target Encoding	[ʁɛsi], [sɛt], [bo], [sis]

- Problem: [ʁɛsi] + [sɛt] = /ʁɛ**sis**ɛt/ which contains the other target [sis].
  This means that a for a child saying 'recit' and 'sept' correctly (or even least partly correctly), the decoder will always score 'six' as correct as [sis] appears in the concatenation /...ʁɛsisɛt/, even if it was mispronounced.

- Solution: As /ɛ/ is the phoneme following the overlapped [sis] in /ʁɛsis**ɛ**t/, we add [sisɛ] to the Error List End of the target 'six'.
  This way when assessing pronunciation of item 'six', Decoder 3 first checks if the error [sisɛ] is contained in a concatenated prediction. If it is found, then /sis/ was probably found due to an overlap, and the score of this target will stay at 0.

9.3.2 Exceptions

To prevent false negatives caused by our decoder's concatenation of successive items, we apply two exceptions:

1. **End-of-Word Silent Letters:** We do not consider potentially erroneous phoneme from a silent letter at the end of a word if it matches the first phoneme of the next item in the reference list.

    - Example: If the reference text is 'finit trou' (/fini tʁu/), then we don't add /finit/ to the error list even if the final /t/ is silent, as is also the first phoneme of 'trou' meaning that the concatenation of these two items (/finitʁu/) contains /finit/.

2. **Beginning-of-Word Silent Letters:** We do not consider a potentially erroneous phoneme from a silent letter at the beggining of an item if it matches the last phoneme of the previous item in the reference list.

    - Example: If the reference text is 'lama août' (/lama ut/), then we don't add /aut/ to the error list for 'août' as it is also the last phoneme of 'lama', and the concatenation of these two items (/lamaut/) contains /aut/.

Decoder 3 effectively addresses the challenges of accurately detecting mispronunciations at item boundaries, reducing the number of false positives, while preserving the robustness to segmentation errors achieved by its predecessor which increased accuracy. Using this decoder on our enhanced pipeline yielded the following results: Using this decoder in our enhanced pipeline yielded the following results for each configuration:

- **Easy configuration:**
    - Accuracy decreased from 77.7% to 77.0% (0.7% decrease)
    - False positive rate decreased from 6.1% to 4.3% (1.8% reduction)
    - Compared to standard decoder: Accuracy increased from 63.3% to 77.0% (13.4% improvement)
- **Complex configuration:**
    - Accuracy improved from 78.8% to 79.6% (0.8% increase)
    - False positive rate decreased from 8.0% to 4.9% (3.1% reduction)
    - Compared to standard decoder: Accuracy increased from 69.7% to 79.6% (9.9% improvement)
- **Pseudo configuration:**
    - Accuracy decreased from 74.4% to 73.7% (0.7% decrease)
    - False positive rate decreased from 6.1% to 4.5% (1.6% reduction)
    - Compared to standard decoder: Accuracy increased from 58.7% to 73.7% (15% improvement)

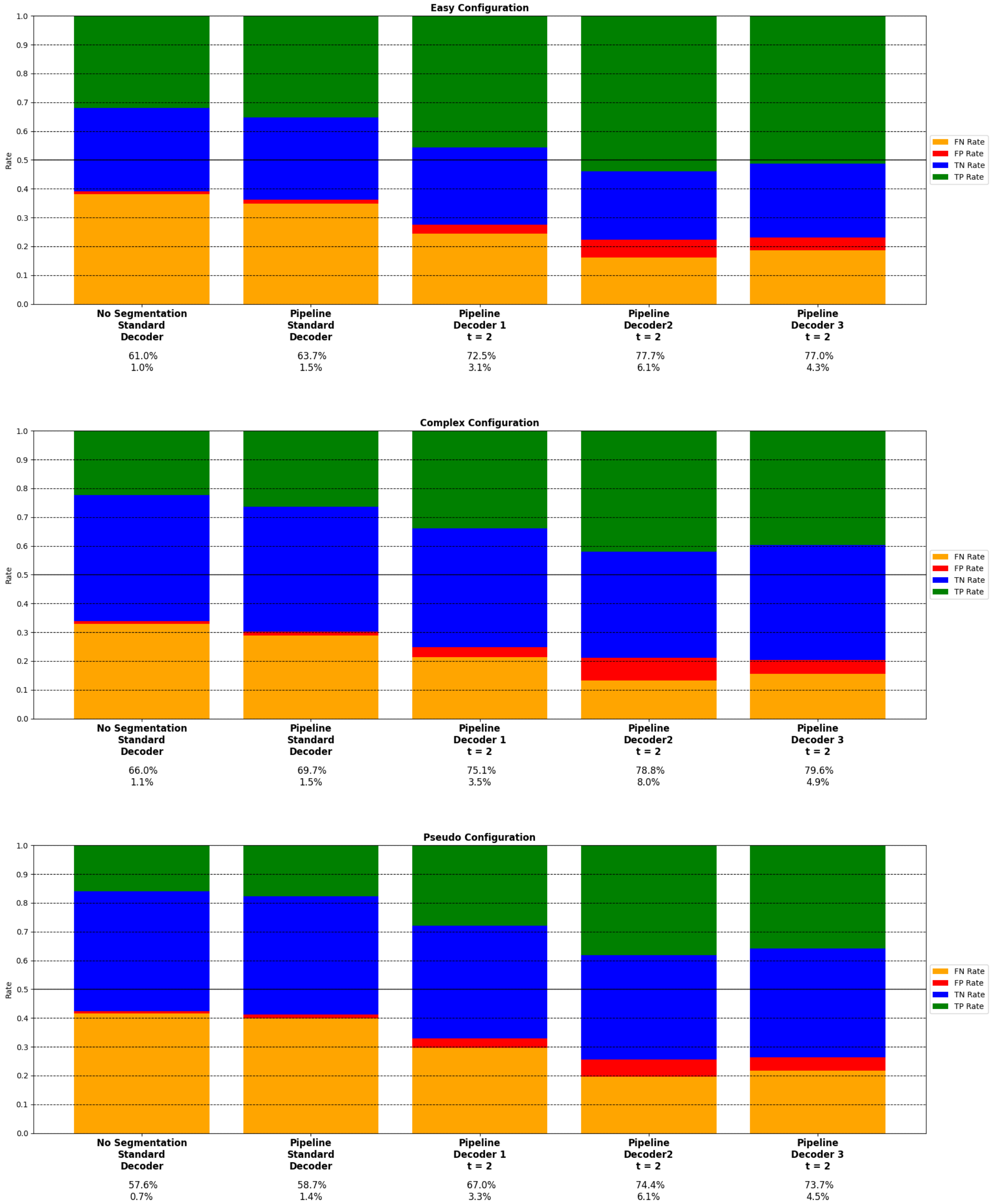


Figure 12: Comparison of the performance of the whole-file processing using the standard decoder (baseline), and the performance the the Standard Wav2Vec2 Decoder, Decoder 1, Decoder 2 and Decoder 3 within the pipeline designed to use ASR as a midpronunciation detection tool. Percentages shown are TP+TN (top) and FP (bottom).

Although there is a very slight decrease in performance compared to its predecessor, Decoder 3 allows us to achieve a significant improvement over the standard decoder (increase of 12.6% in performance) while maintaining false positive rates below our threshold. This represents the best of both worlds: enhanced accuracy and controlled false positives. Although there is a very small decrease in performance compared to its predecessor, Decoder 3 enables us to achieve a significant improvement over the standard decoder while maintaining false positive rates below our threshold. This represents the best of both worlds: enhanced accuracy and controlled false positives. The effectiveness of this approach, particularly in identifying mispronunciations involving silent letters, is demonstrated in the examples provided in Appendix D. The error lists are derived from general linguistic rules rather than statistical patterns of errors from the dataset. This data-agnostic approach enhances the system's generalizability and ease of implementation as the professionals can adapt it to their data using only the speech tasks, thus also mitigating biases that can arise from tuning the errors based limited or skewed datasets.

10 Limitations

In Section 4.1, we discussed the professional scoring mapping from 0,1,2 to our binary correct/incorrect score. While mapping a score of 1 (almost correct) to "correct" works in most cases there are notable cases where discrepancies arise between the scoring method used by the professionals in our dataset and ASR operations, which require careful consideration.

Insertions within an item One challenging scenario involves insertions in the middle of an item. For instance, if a child inserts the phoneme /m/ in 'cou{m}leur', a clinician would score this as 1 (almost correct) according the the scoring protocol. However, the ASR would transcribe it as 'coumleur' and score it as incorrect since it doesn't match the target item 'couleur'. Accounting for all possible insertions across all items is impractical and subjective, as the line between "incorrect" and "almost correct" in this case can be blurry. We found in our dataset 41 scores with this mismatch, accounting for approximately 0.38% of total scores.

Sounding out then incorrect Another discrepancy occurs when an item is first sounded out correctly but then pronounced incorrectly (e.g., 'cou.leur coular'). Clinicians would score this as 0, recognizing the ultimate incorrect pronunciation. However, the ASR would likely recognize the correct word 'couleur' and score it as correct. Implementing a solution for this is challenging, as it's difficult to distinguish whether what is following is a mispronunciation of the current item or something else. We found in our dataset 18 scores with this mismatch, accounting for approximately 0.13% of total scores.

Accentuated pronunciation of the final 'e' in items A third area of mismatch involves the accentuated pronunciation of final 'e' in items. In the clinical scoring system, such cases are scored 1 if the final syllable is not sounded out, and 0 if it is. Our ASR system, robust to sounding out, struggles to reliably distinguish between these subtle pronunciation differences, which are often subject to human interpretation. We found in our dataset 18 scores with this mismatch, accounting for approximately 0.16% of total scores.

While these mismatches represent a small percentage of our current dataset, they highlight fundamental differences between the binary ASR-based scoring and the more nuanced professional scoring methods. These discrepancies stem from the inherent limitations of ASR systems in capturing nuanced aspects of speech that human professionals can more easily interpret.

11 Future Work

While our current approach has demonstrated promising results in automated speech assessment for children, several limitations and areas for potential improvement merit further investigation.

A primary focus for future work is refinement of the Wav2Vec2 Phonemizer model through the use of larger pre-training and fine-tuning datasets. The current model [25] is based on the Wav2Vec2 base model, pre-trained on 22.8k hours of parliament speech and fine-tuned on 941 hours of read speech. A promising avenue for improvement involves leveraging the multilingual Wav2Vec2 XLS model, which has been pre-trained on a dataset 22 times larger and more diverse including read speech and read book corpora. This approach could potentially enhance performance, given the general trend of improved results with larger, more diverse pre-training datasets in speech recognition tasks and the significant mismatch between the dataset used to pretrain the Wav2Vec2 Phonemizer's used in our pipeline and our dataset. Comparative studies using these enhanced models could provide valuable insights into the impact of pre-training data scale and diversity on phoneme recognition accuracy in child speech assessment tasks.

Another potential direction for future study, contingent on data availability, would be to evaluate the current approach on adult speech datasets with comparable tasks. While we currently lack such datasets, a comparative analysis of the system's performance on child versus adult speech could provide insights into the distinct challenges associated with child speech recognition and those related to the specific content being evaluated.

Finally, an important area for improvement lies in addressing the scoring mismatches. While these mismatches currently represent a small percentage of our dataset, developing more sophisticated algorithms to handle these edge cases could further align our automated scoring with professional assessment methods.

12 Conclusion

Our research has led to the development of an enhanced pipeline that strategically combines the strengths of word-level and phoneme-level processing, addressing the limitations observed in state-of-the-art ASR models for our specific task. This system demonstrates significant improvements over the standard Wav2Vec2 Phonemizer performance in assessing children's pronunciation accuracy, particularly in handling the variability of read child speech.

The approach consists of an optimized audio segmentation strategy that addresses the high variability of child speech rhythms and rates by optimizing the amount of context in each resulting segment. Building upon this segmentation, we developed a series of increasingly sophisticated decoding algorithms that progressively tackle key challenges in detecting mispronunciations in child speech. These algorithms address issues such as pronunciation variability, noisy speech, segmentation errors arising from variable pause frequency patterns and rhythms in children's reading speech, and the detection of subtle mispronunciations at item boundaries.

A notable strength of our system lies in its generalizability. It leverages available Wav2Vec2 models fine-tuned on adult speech datasets, which are widely available for many languages. In cases where fine-tuned models are not available for a particular language, the pretrained Wav2Vec2 models can be fine-tuned on adult speech datasets which are more available and easier to obtain compared to child speech datasets, especially in low-resource languages. As an example, the Comon Voice dataset, which was used for finetuning both our Wav2Vec2 Segmenter and Wav2Vec2 Phonemizer, contains read speech in more than a hundred languages. This approach makes our pipeline more adaptable across languages, facilitating its use in diverse linguistic contexts. Moreover, the system's data-agnostic approach allows for quick and easy adaptation and deployement to various datasets and speech evaluation tasks without the need for task-specific data collection or model fine-tuning as the system requires only the phonetic targets of the items to be evaluated. This feature significantly reduces the time and resources required for deployment in new contexts. Furthermore, it allows for quick and precise adjustments to accommodate different regional accents simply by modifying the phonetic encodings of each target, enabling quick and straightforward adaptation. Finally, by adjusting the threshold parameter, the system can be tuned to be more conservative or liberal in its assessments, catering to different speech evaluation task requirements.

Overall, our system's ease of use, rapid deployment without requiring data collection, ability to cater to regional accents, and flexibility in assessment criteria make it a valuable and versatile tool for assessing pronunciation accuracy across a wide range of speech evaluation tasks.

Appendix

A Item-Level Segmentation Strategy and the Importance of Padding

To address these challenges, we explored segmenting audio at the item level using timestamps from the ASR Segmenter, and added padding to mitigate boundary effects. For example, if the timestamps of the pseudo word 'timpunet' are (3.5 s - 4.5 s), a 0.5 s padding would result in a segment from 3.0 s to 5.0 s. We then fed these segments to the phonemizer ASR instead of processing the whole audio file. Interestingly, we observed that the phonetic transcription of a same segment varied depending on the amount of padding added. We tested various padding durations (0.1 s, 0.3 s, 0.5 s, 0.7 s, and 1 s) and observed that accuracy improved as the padding increased as illustrated in Figure 13. Specifically:

1. **Easy configuration:**
   - 12% increase from 0.1s to 0.5s padding, and 8.2% increase from 0.5s to 1s padding
   - Achieved 62.8% accuracy (1.8% improvement over whole-file processing)
2. **Complex configuration:**
   - 8.4% increase from 0.1s to 0.5s padding, and 6% increase to 1s padding
   - Achieved 69.4% accuracy (3.4% improvement over whole-file processing)
   - Only configuration achieving similar performance with less than 1s padding (0.7s)
3. **Pseudo configuration:**
   - 4.5% increase from 0.1s to 0.5s padding, and 6% increase to 1s padding
   - Achieved 56.4% accuracy (0.8% decrease compared to whole-file processing)

Comparing item-level segmentation with whole-file processing revealed several challenges illustrated in Tables 7 & 8:

Phoneme Omissions:

- Shorter paddings often resulted in an increase of omitted phonemes, especially at the boundaries. For example, in Figure 7, the item 'sidomelle' was transcribed as /tomɛl/ with 0.1 s padding, omitting the initial phonemes /si/. Moreover both the 0.1 s and 0.5 s padding transcription misrecognized the phoneme /d/ as /t/.

- Whole-file transcription also shows occurences of phoneme omissions, especially vowels within an item. In Figure 8, the word 'régal' was transcribed as /twaʁgɑl/ in the whole segment transcription, omitting the phoneme /e/.

Item Boundary Delimitation:

- Shorter padding tends to be much more sensitive to pauses, often splitting an item into two. For instance, in Table 7, the item 'firoie' was transcribed as /fi wa/ with 0.1s padding, effectively splitting it into two parts, while the whole-segment processing kept it intact as /fiʁwa/.

- Whole-segment processing also encountered issues with splitting items into two. In Table 8, the item 'tiveau' was transcribed as /divo/ in the whole segment, splitting the item incorrectly, while in the item-level segmentation with 1s padding, it was correctly transcribed as /divo/. Additionally, whole-segment processing shows a tendency to merge some consecutive items, as seen in the incorrect transcription of 'toit régal' as /twaʁgɑl/.

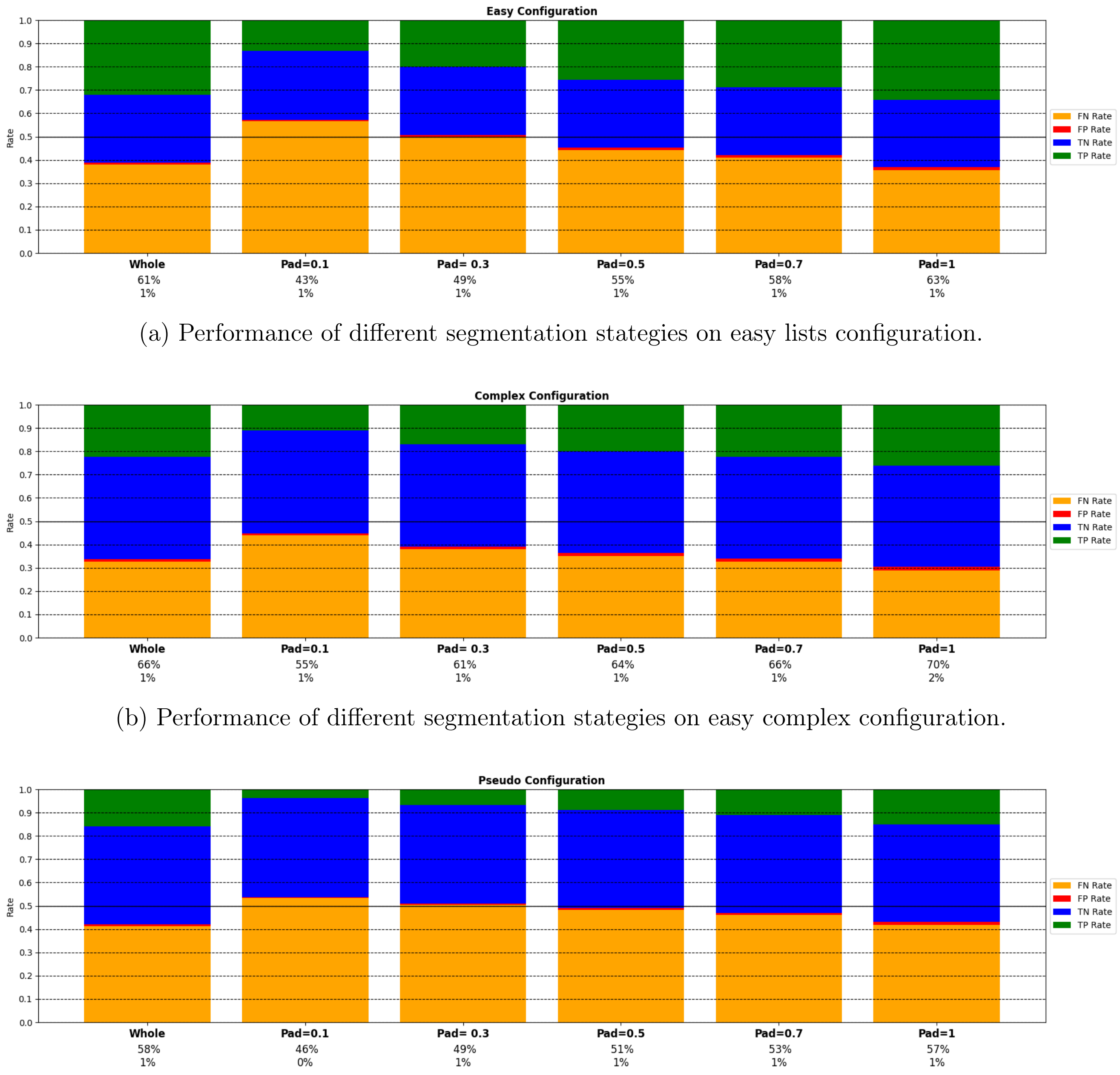
Reference:	'sidomelle tiveau firoie...'
Whole segment:	/sidomɛl di vo fiʁwa .../
Item + 0.1s pad:	/tomɛl/, /si/, /fi wa/, ...
Item + 0.5s pad:	/sitomɛl/, /i fo/, /fi ʁwa/, ...
Item + 1s pad:	/sidomɛl divo/, /omɛl til fo/, /fiʁwa ti/, ...

Table 7: Phoneme transcription with various padding settings and for the whole audio file.

Reference:	"toit régal..."
Whole segment:	/twaʁgal .../
Item + 0.1s pad:	/də vwaʁ/, /ʁekal/, ...
Item + 0.5s pad:	/twa ʁe/, /vwa ʁegal/, ...
Item + 1s pad:	/twa ʁeg/, /twa ʁegal ʒ/, ...

Table 8: Phoneme transcription with various padding settings and for the whole audio file.

Overall, adding 1 second of padding mitigates challenges seen with shorter padding, as the additional context before and after the word helps reduce boundary effects and improves phoneme accuracy at the borders and increased overall performance by 1.4%. By processing the whole audio file segment by segment with a good padding value, we observe an improvement in the aforementioned issues encountered when processing the whole audio file, namely word boundaries and phoneme omissions. The complex configuration benefits most from this approach, likely due to the presence of longer words in these lists, resulting in slightly longer segments. This suggests that finding the right balance between too much and too little audio context as input to the Wav2Vec2 Phonemizer is crucial for optimal performance.



(a) Performance of different segmentation stategies on easy lists configuration.

(b) Performance of different segmentation stategies on easy complex configuration.

(c) Performance of different segmentation stategies on pseudo lists configuration

Figure 13: Comparison of transcription accuracy for various segmentation strategies. The plots show performance for whole-file processing, and item-level segmentation with different padding durations (0.1s to 1s). The two percentages shown under each bar corresponds to the percentage of TP+TN (top) and the percentage of FP (bottom)

B From the Standard Wav2Vec2 Decoder to Decoder 1

B.1 Example A: Other Very Probable Outputs

Consider the following scenario:

Reference Text	'lama poids mille'
Target Encoding	[lama], [pwa], [mil]
Child Utterance	/lama pwa mil/
Ground Truth Score	[1, 1, 1]

B.1.1 Wav2Vec2 Standard Decoder : Limitations Illustrated

Figure 14 illustrates the logit matrix produced by the Wav2Vec2 Phonemizer. For simplicity, we've included only a subset of the 59 tokens in our representation and removed the successive repetitions of timeframes. Note that the token '|' corresponds to a space character.

- The green cells in the matrix highlight the highest logit values for each timeframe. Following the standard decoding process, this results in the output */lama bwa mil/*. Consequently, the accuracy will be scored as [**1**, **0**, **1**].

- However, logit value of the phoneme /p/ (blue cell) is very close to the value of phoneme /b/ for the 6th timeframe. This implies that */lama pwa mil/* is also a very probable output, but standard decoder's approach doesn't account for these near-matches.

This example underscores the standard Wav2Vec2 decoder failing to consider the confidence levels of alternative phonemes resulting in an incorrect assessement of the child pronunciation.

B.1.2 Decoder 1: Handling the limitations

As Decoder 1 needs Wav2Vec2 Phonemizer output to get the space characters offsets, we updated the scenario:

Reference Text	'lama poids mille'
Target Encoding	[lama], [pwa], [mil]
Child Utterance	/lama pwa mil/
Ground Truth Score	[1, 1, 1]
Phonemizer's Output	/lama pwa mil/

Figure 15 shows the logit matrix for this segment, processed by our new decoding strategy. Let's walk through the process:

1. **Initial Segmentation**: We have space character ('|'), resulting in 3 submatrices (for subsegments /lama / , / pwa / , / mil/)

2. **Multiple Candidate Selection**: For each timeframe within a submatrix, we select at most k phonemes whose logit values are within our threshold t of the top candidate. In the example:

   - Green cells represent the highest logit values for each timeframe.
   - Blue cells indicate additional phonemes selected based on our threshold criterion ($t = 2$). For each timeframe, we select up to k additional phonemes:
     - Submatrix for item /lama /: No additional phonemes selected.
     - Submatrix for item / pwa /: Phoneme /p/ is selected for timeframe 2, and phoneme /i/ is selected for timeframe 4.
     - Submatrix for item / mil/: Phoneme /a/ is selected for timeframe 2, and phoneme /m/ is selected for timeframe 3.
   - Purple cells represent logit values close to the highest values but not selected based on the threshold criterion. For a phoneme with a purple cell, if the difference between the purple cell's logit value and the maximum value for that timeframe is greater than $t = 2$, it is not considered.

3. **Combinatorial Prediction Generation**: For each subsegment, we generate at most M combinations of the selected phonemes. This results in the following possibilities outputs for each item:

   ```
   [/lama/] [/bwa/, /pwa/, /bwi/, /pwi/] [/mim/, /mam/, /mil/, /mal/]
   ```

4. **Target Matching**: These generated combinations are compared against the target phonetic encoding [lama], [pwa], [mil]

   - 'lama': [lama] found in [/lama/]
   - 'poids': [pwa] found in [/bwa/, /pwa/, /bwi/, /pwi/]
   - 'mille': [mil] found in [/mim/, /mam/, /mil/, /mal/]

5. **Score: [1, 1, 1]** → The decoder now correctly identifies all targets despite characters.

While for the second item the standard decoder would only consider the output /bwa/, our new decoder considers /pwa/ as a valid possibility. This demonstrates how Decoder 1 can capture pronunciation nuances that the standard decoder might miss.

a	b	d	e	f	h	i	j	k	l	m	n	o	p	r	s	t	u	w	x	y	z	ʁ	g	ɛ	~	|
									🟩																	
🟩																										
										🟩																
🟩																										
																										🟩
	🟩												🟦													
																		🟩								
🟩																										
																										🟩
										🟩																
						🟩																				
									🟩																	

Figure 14: Simplified logit matrix outputted by Wav2Vec2 Phonemizer

a	b	d	e	f	h	i	j	k	l	m	n	o	p	r	s	t	u	w	x	y	z	ʁ	g	ɛ	~	|
									🟩																	
🟩																										
										🟩																
🟩																										
																										🟩

																										🟩
	🟩 9												🟦 8			🟪 6										
																		🟩								
🟩						🟦																				

																										🟩
										🟩																
🟦 3						🟩 4																				
									🟩 6	🟦 7																

Figure 15: Simplified logit matrix logit matrix processing of Decoder 1

C From Decoder 1 to Decoder 2

C.1 Example B: Missing Space In Wav2Vec2 Phonemizer's Outputs

Consider the following scenario:

Reference Text	'lama poids mille'
Target Encoding	[lama], [pwa], [mil]
Child Utterance	/lama pwa mil/
Ground Truth Score	[**1**, **1**, **1**]
Phonemizer's Output	/lama pwamil/ (missing space between /pwa/ and /mil/)

C.1.1 Decoder 1: Limitations Illustrated

1. **Initial Segmentation**: We have 1 space character, resulting in two submatrices, one for subsegment /lama / and one for / pwamil/
2. **Multiple Candidate Selection & Combinatorial Prediction Generation**:

```
[/lama/] [/bwamim/, /bwamam/, /bwamil/, /bwamal/, /bwimim/,
            /bwimam/, /bwimil/,..., /pwamam/, /pwamil/, /pwamal/]
```

3. **Target Matching**: Target phonetic encoding: [lama], [pwa], [mil]
   - 'lama': Match for [lama] found in [/lama/]
   - 'poids': No match for target [pwa]
   - 'mille': No match for target [mil].
4. **Score:** [**1**, **0**, **0**] → The decoder misidentifies the targets [pwa] and [mil] as incorrect due to a missing space, even though it was correctly pronounced.

C.1.2 Decoder 2: Handling the limitations

1. **Initial Segmentation**: /lama / , / pwamil/
2. **Multiple Candidate Selection and Combinatorial Prediction Generation**:

```
[/lama/] [/bwamim/, /bwamam/, /bwamil/,,..., /pwamil/, /pwamal/]
```

3. **Concatenation of Adjacent Predictions**: Each prediction of the first item is concatenated with each prediction of the second item:

```
=> [/lama/] + [/bwamim/,..., /pwamal/] = [/lamabwamim/,...,
                                          /lamapwamil/,...]
```

4. **Flexible Matching**: Target phonetic encoding: [lama], [pwa], [mil]
    - 'lama': [lama] contained in concatenated predictions
    - 'poids': [pwa] contained in concatenated predictions
    - 'mille': [mil] contained in concatenated predictions
5. **Score: [1, 1, 1]** → The decoder correctly identifies all targets despite the missing space.

C.2 Example C: Extra Spaces In Wav2Vec2 Phonemizer Outputs

C.2.1 Decoder 1: Limitations Illustrated

Reference Text	'lama poids mille'
Target Encoding	[lama], [pwa], [mil]
Child Utterance	/lama pwa mil/
Ground Truth Score	[**1, 1, 1**]
Phonemizer's Output	/lama pw a mil/ (extra space within item /pwa/)

1. **Initial Segmentation**: /lama /, / pw /, / a /, / mil/
2. **Multiple Candidate Selection & Combinatorial Prediction Generation**:

    ```
    [/lama/] [/bw/, /pwi/] [/a/, /i/] [/mim/, /mam/, /mil/, /mal/]
    ```

3. **Target Matching**: Target phonetic encoding: [lama], [pwa], [mil]
    - 'lama': [lama] found in [/lama/]
    - 'poids': No match for targets [pwa].
    - 'poids': [mil] found in [/mim/, /mam/, /mil/, /mal/]
4. **Score:** [**1, 0, 1**] → The decoder misidentifies the target [pwa] as incorrect due to a missing space, even though it was correctly pronounced.

C.2.2 Decoder 2: Handling the limitations

1. **Initial Segmentation**: /lama/ , /pw/ , /a/ , /mil/
2. **Multiple Candidate Selection and Combinatorial Prediction Generation**:

    ```
    [/lama/] [/bw/, /pw/] [/a/, /i/] [/mim/, /mam/, /mil/, /mal/]
    ```

3. **Concatenation of Adjacent Predictions**:

```
=> [/lama/] + [/bw/, /pw/] + [/a/, /i/]
    = [/lamabwa/,/lamabwi/, /lamapwa/, /lamapwi/]
=> [/bw/, /pw/] + [/a/, /i/] + [/mim/, /mam/, /mil/, /mal/]
    = [/bwamim/, /bwimim/, ..., /pwamil/,..., /pwimal/]
```

4. **Flexible Matching**: Target phonetic encoding: [lama], [pwa], [mil]
   - 'lama': [lama] contained in first concatenated list
   - 'poids': [pwa] contained in both concatenated lists
   - 'mille': [mil] contained in second concatenated list
5. **Score: [1, 1, 1]** → The decoder correctly identifies all targets despite the extra space.

D From Decoder 2 to Decoder 3

D.1 Example D: Silent Letters at Item Boundaries

To illustrate the limitations of Decoder 2, let's consider a scenario where the target phrase contains both initial and final silent letter challenges.

Reference Text	'poids mille août'
Target Encoding	[lama], [pwa], [ut]
Child Utterance	/pwad mil aut/
Ground Truth Score	[0, 1, 0]
Phonemizer's Output	/pwad mil aut/

D.1.1 Decoder 2: Limitations Illustrated

1. **Initial Segmentation**: /pwad/ , /mil/ , /aut/
2. **Multiple Candidate Selection & Combinatorial Prediction Generation**:

```
[/pwad/, /pwat/] [/mil/, /mim/] [/aut/, /ut/]
```

3. **Concatenation of Adjacent Predictions**:

```
[/pwadmimaut/, /pwadmimout/, /pwadmilaut/, /pwadmilout/,
/pwatmimaut/, /pwatmimout/, /pwatmilaut/, /pwatmilout/]
```

4. **Flexible Matching**: Target phonetic encoding: [pwa], [mil], [ut]
   - 'poids': [pwa] contained at least a concatenation
   - 'mille' [mil] contained in at least a concatenation

   - 'août' [ut] contained in at least a concatenation

5. **Score: [1, 1, 1]** → The decoder incorrectly marks both 'poids' and 'août' as correct.

Decoder 2's containment-based matching strategy is unable to distinguish between correct pronunciations if the error is made by inserting phonemes before and after an item.

- For 'poids': Decoder 2 fails to detect the mispronunciation of the final 'd' because the correct pronunciation /pwa/ is still contained within the concatenated prediction /pwad/
- For 'août': Decoder 2 fails to detect the mispronunciation of the initial 'a' because the correct pronunciation /ut/ is still contained within the concatenated prediction /aut/.

D.1.2 Decoder 3: Handling the limitations

As Decoder 3 needs error lists for each target output, we updated the scenario:

Reference Text	'poids mille août'
Target Encoding	[lama], [pwa], [ut]
Error Lists End	[], [pwad], []
Error Lists Beginning	[], [], [aut]
Child Utterance	/pwad mil aut/
Ground Truth Score	[0, 1, 0]
Phonemizer's Output	/pwad mil aut/

1. **Initial Segmentation**: /pwad/ , /mil/ , /aut/
2. **Multiple Candidate Selection & Combinatorial Prediction Generation**:

   ```
   [/pwad/, /pwat/] [/mil/, /mim/] [/aut/, /ut/]
   ```

3. **Concatenation of Adjacent Predictions**:

   ```
   [/pwadmimaut/, /pwadmimout/, /pwadmilaut/, /pwadmilout/,
   /pwatmimaut/, /pwatmimout/, /pwatmilaut/, /pwatmilout/]
   ```

4. **Error Check Before Containment Check**:
   - 'poids': [pwad] contained in at least a concatenation
   - 'mil': no errors lists
   - 'août': [aut] contained in at least a concatenation
5. **Flexible Matching**:

- 'poids': error found
- 'mille': [mil] contained in half of the concatenations
- 'août': error found

6. **Score: [0, 1, 0]** → The decoder correctly assesses pronuncation of each target.

References

[1] Y. Sunil, S. M. Prasanna, and R. Sinha, "Children's Speech Recognition Under Mismatched Condition: A Review," *IETE Journal of Education*, vol. 57, pp. 96–108, July 2016.

[2] S. Lee, A. Potamianos, and S. Narayanan, "Acoustics of children's speech: developmental changes of temporal and spectral parameters," *The Journal of the Acoustical Society of America*, vol. 105, pp. 1455–1468, Mar. 1999.

[3] J. E. Huber, E. T. Stathopoulos, G. M. Curione, T. A. Ash, and K. Johnson, "Formants of children, women, and men: The effects of vocal intensity variation," *The Journal of the Acoustical Society of America*, vol. 106, pp. 1532–1542, Sept. 1999.

[4] A. Potamianos and S. Narayanan, "Spoken dialog systems for children," in *Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181)*, vol. 1, pp. 197–200 vol.1, May 1998. ISSN: 1520-6149.

[5] M. Gerosa, D. Giuliani, and F. Brugnara, "Acoustic variability and automatic recognition of children's speech," *Speech Communication*, vol. 49, pp. 847–860, Oct. 2007.

[6] V. Hazan and M. Pettinato, "The emergence of rhythmic strategies for clarifying speech: Variation of syllable rate and pausing in adults, children and teenagers," May 2014.

[7] P. Gurunath Shivakumar and S. Narayanan, "End-to-end neural systems for automatic children speech recognition: An empirical study," *Computer Speech & Language*, vol. 72, p. 101289, Mar. 2022.

[8] D. Giuliani and M. Gerosa, "Investigating recognition of children's speech," in *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings. (ICASSP '03).*, vol. 2, pp. II–137, Apr. 2003. ISSN: 1520-6149.

[9] P. Gurunath Shivakumar and P. Georgiou, "Transfer learning from adult to children for speech recognition: Evaluation, analysis and recommendations," *Computer Speech & Language*, vol. 63, p. 101077, Sept. 2020.

[10] J. Zuluaga-Gomez, A. Prasad, I. Nigmatulina, S. Sarfjoo, P. Motlicek, M. Kleinert, H. Helmke, O. Ohneiser, and Q. Zhan, "How Does Pre-trained Wav2Vec 2.0 Perform on Domain Shifted ASR? An Extensive Benchmark on Air Traffic Control Communications," Oct. 2022. arXiv:2203.16822 [cs, eess].

[11] S. Shraddha, J. L. G, and S. K. S, "Child Speech Recognition on End-to-End Neural ASR Models," in *2022 2nd International Conference on Intelligent Technologies (CONIT)*, pp. 1–6, June 2022.

[12] S. P. Dubagunta, S. Hande Kabil, and M. Magimai.-Doss, "Improving Children Speech Recognition through Feature Learning from Raw Speech Signal," in *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5736–5740, May 2019. ISSN: 2379-190X.

[13] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust Speech Recognition via Large-Scale Weak Supervision," Dec. 2022. arXiv:2212.04356 [cs, eess].

[14] J. C. Vásquez-Correa and A. Álvarez Muniain, "Novel Speech Recognition Systems Applied to Forensics within Child Exploitation: Wav2vec2.0 vs. Whisper," *Sensors (Basel, Switzerland)*, vol. 23, p. 1843, Feb. 2023.

[15] M. Fuckner, S. Horsman, P. Wiggers, and I. Janssen, "Uncovering Bias in ASR Systems: Evaluating Wav2vec2 and Whisper for Dutch speakers," in *2023 International Conference on Speech Technology and Human-Computer Dialogue (SpeD)*, (Bucharest, Romania), pp. 146–151, IEEE, Oct. 2023.

[16] M. Bain, J. Huh, T. Han, and A. Zisserman, "WhisperX: Time-Accurate Speech Transcription of Long-Form Audio," in *INTERSPEECH 2023*, pp. 4489–4493, ISCA, Aug. 2023.

[17] A. Baevski, H. Zhou, A. Mohamed, and M. Auli, "wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations," Oct. 2020. arXiv:2006.11477 [cs, eess].

[18] R. Lu, M. Shahin, and B. Ahmed, "Improving Children's Speech Recognition by Fine-tuning Self-supervised Adult Speech Representations," Nov. 2022. arXiv:2211.07769 [cs, eess].

[19] R. Jain, A. Barcovschi, M. Y. Yiwere, D. Bigioi, P. Corcoran, and H. Cucu, "A WAV2VEC2-Based Experimental Study on Self-Supervised Learning Methods to Improve Child Speech Recognition," *IEEE Access*, vol. 11, pp. 46938–46948, 2023.

[20] A. Barcovschi, R. Jain, and P. Corcoran, "A comparative analysis between Conformer-Transducer, Whisper, and wav2vec2 for improving the child speech recognition," Nov. 2023. arXiv:2311.04936 [cs, eess].

[21] R. Jain, A. Barcovschi, M. Yiwere, P. Corcoran, and H. Cucu, "Adaptation of Whisper models to child speech recognition," July 2023. arXiv:2307.13008 [cs, eess].

[22] "jonatasgrosman/exp_w2v2t_fr_vp-100k_s973 · Hugging Face," July 2023.

[23] J. Grosman, "Fine-tuned XLS-R 1B model for speech recognition in French," 2022.

[24] A. Babu, C. Wang, A. Tjandra, K. Lakhotia, Q. Xu, N. Goyal, K. Singh, P. von Platen, Y. Saraf, J. Pino, A. Baevski, A. Conneau, and M. Auli, "XLS-R: Self-supervised Cross-lingual Speech Representation Learning at Scale," Dec. 2021. arXiv:2111.09296 [cs, eess].

[25] M. Olivier, J. Hauret, and Bavu, "wav2vec2-french-phonemizer (Revision e715906)," 2023.

[26] M. Nakamura, K. Iwano, and S. Furui, "Differences between acoustic characteristics of spontaneous and read speech and their effects on speech recognition performance," *Computer Speech Language*, vol. 22, no. 2, pp. 171–184, 2008.

[27] F. Vassiliki and A. Potamianos, “Linguistic Analysis of Spontaneous Children Speech,” 2008.

[28] P. Howell and K. Kadi-Hanifi, “Comparison of prosodic properties between read and spontaneous speech material,” *Speech Communication*, vol. 10, no. 2, pp. 163–169, 1991.