Phonotomizer: A Compact, Unsupervised, Online Training Approach to Real-Time, Multilingual Phonetic Segmentation

Michael S. Yantosca, Albert M. K. Cheng
Department of Computer Science
University of Houston
Houston, TX, USA
msyantosca@uh.edu, amcheng@uh.edu

Abstract

Phonetic transcription requires significant time and expert training. Automated, state-of-the-art text-dependent methods still involve substantial pre-training annotation labor and may not generalize to multiple languages. Hallucination of speech amid silence or non-speech noise can also plague these methods, which fall short in real-time applications due to *post hoc* whole-phrase evaluation. This paper introduces Phonotomizer, a compact, unsupervised, online training approach to automatic, multilingual phonetic segmentation, a critical first stage in transcription. Unlike prior approaches, Phonotomizer trains on raw sound files alone and can modulate computational exactness. Preliminary evaluations on Irish and Twi, two underrepresented languages, exhibit segmentation comparable to current forced alignment technology, reducing acoustic model size and minimizing training epochs.

1 Introduction

Computer scientists and linguists have spent decades honing automated speech-to-text (STT) systems to drive interfaces through natural language processing (NLP). At the NLP pipeline head, speech segmentation tokenizes continuous acoustic data into chunks for downstream processing. Both layered phonic analysis and direct word correlation depend on segmenter accuracy and speed. Many mistakes in final STT output stem from subtle segmenter perturbations (Hamooni and Mueen, 2014).

Intuitive user experience strongly motivates research in fields as disparate as accessibility, transcription, and telecommunication. As the global economy grows to depend on computing, natural interfaces to these systems can reduce barriers to adoption for diverse language communities.

Computers can struggle to process speech with unfamiliar accents, dialects, languages, or noise. Even in clean-room settings, speaker variation may impair consistency. Consecutive sounds may alter each other, e.g., by tongue movement in palatalization or throat constriction in pharyngealization. Acoustic variability demands flexibility.

Existing STT systems tend to fall into two camps: hierarchical and end-to-end. Hierarchical approaches, e.g., SegFeat (Kreuk et al., 2020), rely on linguistic concepts with acoustic models often tailored to one language. Modern end-to-end systems, e.g., DeepSpeech (Hannun et al., 2014), apply data-, time-, and energy-hungry deep learning (DL) techniques to map orthography directly from raw acoustic data but often struggle with unforeseen words. Broken dependencies and abandonware litter the field in both categories.

This work remedies these shortcomings with a compact tool that trains online with minimal third-party dependencies. Its phonetic approach supports adaptive, mixed-criticality real-time systems balancing accuracy and response time through inexact computation and fosters a sustainable, hierarchical approach to STT systems by leaving sufficient headroom for downstream applications.

Phonotomizer contributes the following improvements to the practice of phonetic segmentation with performance comparable to the state-of-the-art in initial experiments:

- reduction in acoustic model size
- meaningful cluster labeling coded for prefix-searchable similarity and explainability
- generality over multiple languages with no need for *a priori* pronunciation dictionaries
- extensibility through interchangeable audio processing pipeline components
- suitability for real-time, online learning

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 12135–12147

2 Related Work

Current STT approaches occupy continua along two axes: training, from supervised to unsupervised, and segmentation, from phonetic/phonemic (by sound) to lexemic (by word).

2.1 Supervised Segmenter/Classifiers

Supervised methods demand *a priori* truth labels and copious training data, which may hamper cataloguing of low-resource and endangered languages. Requisite manual annotation compounds high time and energy costs, and standard datasets in the literature cover few languages (e.g., Garofolo et al., 1992; Pitt et al., 2005; Panayotov et al., 2014). A significant body of work (e.g., Lin and Wang, 2022; Taguchi and Chiang, 2024; El Kheir et al., 2024; Liu et al., 2024; Fu et al., 2024) extends Wav2Vec (Schneider et al., 2019; Baevski et al., 2020) and executes experiments on well-studied languages. A recurrent neural network (RNN) devised by Kreuk et al. (2020) transferred from English to Hebrew with $\approx 10\%$ loss, but lack of textual ground truth may impair greenfield linguistic studies.

McAuliffe et al. (2017) debuted the Montreal Forced Aligner (MFA) to time-correlate audio with transcriptions via a sliding triphone (3-sound) window for better auditory context. Though citing wide multilingual support, the initial paper only evaluated North American English. Training new acoustic models requires set pronunciation dictionaries on top of recordings and orthographic text.

2.2 Unsupervised Segmenter/Classifiers

Unsupervised approaches classify sound without the aid of ground truths. Key issues are cluster fragmentation, speaker dependence, and consistency. Typical approaches include k-means clustering (Duda et al., 2001; Bhati et al., 2018), RNNs (Wu et al., 2021), Bayesian Gausian Mixture Models (GMMs) (Kamper et al., 2016; Kamper, 2019), and adversarial deep neural networks (DNNs) (Tsuchiya et al., 2018). Some studies attempt to model infant language acquisition, aiming for simultaneous word and phone discovery and multi-modal reinforcement (Taniguchi et al., 2016; Tada et al., 2017; Okuda et al., 2022; Taniguchi et al., 2023). As with the supervised approaches, the focus of study tends to favor languages with large corpora and comprehensive transcription.

2.3 Real-Time Classifiers

Baruah et al. (2023) explored so-called "I Don't Know" (IDK) classifiers under time constraints, focusing on pre-trained, readily interchangeable DNNs. Optimally ordered IDK classifier cascades may save time on average, but the entire cascade must meet deadlines in the worst case. Nguyen et al. (2024) used probabilistic analysis to impart dynamism to the cascade and skip intermediate stages for up to 17% quicker response times.

2.4 Segmentation by Phone/Phoneme (Sound)

Systems that segment by phone or phoneme build a hierarchical representation based on linguistic theory. By phoneme implies an accord with a given language's phoneme inventory; by phone, a cross-lingual sound space. Most techniques take a phonemic approach to match available transcription resources (Wang et al., 2015; Gao et al., 2020). While linguistically robust, this granularity can induce latency to orthographic output.

Difficulties in phoneme identification have led some to shift to lexemic approaches (Gao et al., 2020). Others have sought to transfer supervised phonemic pre-training from high- to low-resource languages (Riviere et al., 2020; Conneau et al., 2021). However, they chiefly evaluated Turkic and Indo-European languages with phonemic ground truths generated by Phonemizer (Bernard and Titeux, 2021) from extant lexemic transcriptions. The studies' "low-resource" languages have far greater representation among the Mozilla Common Voice datasets (The Mozilla Foundation, 2022) than, say, Twi, a variant of Akan spoken in Ghana.

2.5 Segmentation by Lexeme (Word)

Scalable neural networks have increased the appeal of direct word segmentation. Proponents claim reduced latency and better noise tolerance, but vocabulary gaps pose challenges.

Abandoning "even the concept of a 'phoneme'", Hannun et al. (2014) presented Deep Speech. Highly scalable, effective lexemic approaches (e.g., Chen et al., 2019; Radford et al., 2023; Barrault et al., 2023a; Barrault et al., 2023b), own this and the Transformer architecture (Vaswani, 2017) as their intellectual predecessors.

However, heavy text dependency presupposes a written form which may not exist. Billion- or trillion-parameter models also incur high ecological tolls (Yu et al., 2024; Luccioni et al., 2024).

3 System Design and Implementation

Phonotomizer seeks to minimize errors and spare resources with universal acoustic features and a compact footprint. Its one-pass k-means approach and adaptivity to deadline pressure suit real-time, mixed-criticality tasks. Built for commodity CPUs and trained on raw sound files alone, Phonotomizer could serve in embedded field linguistics contexts under sparse network access and tight resource constraints, reaping a windfall in its inherent amenity to data privacy versus cloud-driven solutions.

Phonotomizer corrects oversights in the original design of Yantosca's (2019) work on ARTIC, an adaptive, real-time audio processing framework which meets real-time deadlines by modulating computational complexity. Work is divided into successive stages of a cascading pipeline.

3.1 Phonotomizer Data Flow

Data flows in Phonotomizer's pipeline as follows: audio ingestion, one-zero gammatone filtration (OZGF) per Lyon (1996), discrete energy separation algorithm (DESA) per Maragos et al. (1993), band estimation per Potamianos and Maragos (1995) and Shokouhi and Hansen (2017), and finally segmentation and classification per Fig. 1. Each pipeline stage spawns a thread, with pipeline management on the program's main thread.

3.1.1 Audio Ingestion

The audio ingestor resamples audio data into non-overlapped frames of N_t 32-bit little endian float (`F32_LE`) raw samples at an internal rate r. By default, $N_t = 2048$ samples, and $r = 48$ kHz.

When reading MP3 files as input, the ingestor skips a priming prelude of initial samples to cohere with ground truth alignments manually transcribed in Praat (Boersma and Weenink, 2023a). Praat skips the first 529 (decoder delay) + 96 (encoder delay) samples (Boersma and Weenink, 2023b). The ingestor multiplies these magic numbers by the ratio of r to the source file's sample rate to achieve the same effect.

3.1.2 OZGF Bank

The OZGF bank divides each ingestor output frame by band-pass filters into B audio spectrum bands. By default, $B = 32$. Band spacing may follow one of these equations:

$$F_{lin} = \left\{ \frac{b + 0.5}{2B} \,\middle|\, b \in [0, B-1] \right\} \tag{1}$$

$$F_{log} = \left\{ \frac{1}{\sqrt{2}} \left[\exp\left(\frac{b + 0.5}{2B} \right) - 1 \right] \,\middle|\, b \in [0, B-1] \right\} \tag{2}$$

$$F_{quad} = \left\{ 2\left(\frac{b + 0.5}{2B} \right)^2 \,\middle|\, b \in [0, B-1] \right\} \tag{3}$$

Linear spacing offers balance while logarithmic and quadratic progressively favor the low end.

The gammatone filters biomimetically replicate cochlear response, suiting speech recognition well. A cascade of O (default = 4) biquad filters per Redmon (2012) implements the OZGF with no automatic gain control, following the "without-AGC open loop" case by Katsiamis et al. (2009).

To govern the filter response shape, a quality factor Q_b is calculated per band according to the following formula where f_c is the band center frequency and f_l and f_h are the lower and upper bounds of the band, respectively (Lyon, 1996):

$$Q_b = \max\left(3, \sqrt{\frac{f_c^2 (10^{\frac{1}{O}} - 1)}{(f_h - f_l)^2}}\right) \tag{4}$$

This approximates the quality factor required to cover the band of interest. To arrive at an analytical solution for Q, a term $1/Q^2$ has been elided from a rearrangement of the bandwidth equation given by Lyon (1996). At higher frequencies, this asymptotically approaches zero. However, this elision may result in gaps between consecutive bands on the lower end of the spectrum.

Before filtration, raw samples pass through a frame mean subtractor per Gangamohan and Gangashetty (2019), who use the technique to clean data for fundamental frequency (F_0) extraction. Similar improvements manifested during development for formant detection and segmentation more generally. In fact, a lower band by itself typically yielded a good estimate of F_0 for $B \geqslant 80$ when $r = 16\,000$ with voicing present.

3.1.3 DESA Calculation

The DESA stage decomposes the audio frames into amplitude modulation (AM) and frequency modulation (FM) components, respectively $a_b(n)$ and $f_b(n)$. This simple method adapts quickly to speech non-stationarity. Maragos et al. (1993) define the Teager-Kaiser energy operator by Eq. (5) and the consecutive sample delta by Eq. (6).

$$\Psi[z(n)] \triangleq z^2(n) - z(n-1)z(n+1) \tag{5}$$

$$\dot{y}_b(n) \triangleq y_b(n) - y_b(n-1) \tag{6}$$

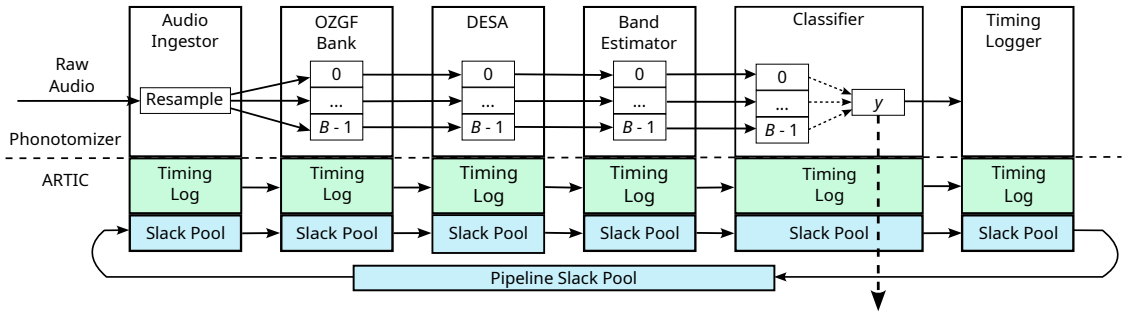

Figure 1: Phonotomizer Pipeline Architecture. A gammatone filterbank divides ingested audio into bands. DESA computes instantaneous amplitude and frequency components for each band. Band estimation averages the DESA output over a window to create dominant frequency estimates with confidence bandwidths, a.k.a., a pyknogram (Potamianos and Maragos, 1995), which informs segmentation and classification. Each classification y of an audio segmentation is immediately recorded. The timing logger stage records stage micro-benchmarks with each cycle. Deadline slack is granted altruistically by each stage and circulated systolically through the system with the data. Stages may claim this slack as needed. Unclaimed deadline slack is recirculated by the pipeline from tail to head.

Per Maragos et al. (1993), the system supports CESA (Eq. (7), Eq. (8)), DESA-1 (Eq. (9), Eq. (10)), and DESA-2 (Eq. (11), Eq. (12)).

$$f_b(n) = \frac{1}{2\pi}\left(\sqrt{\frac{\Psi[\dot{x}_b(n)]}{\Psi[x_b(n)]}}\right) \tag{7}$$

$$|a_b(n)| = \frac{\Psi[x_b(n)]}{\sqrt{\Psi[\dot{x}_b(n)]}} \tag{8}$$

$$f_b(n) = \frac{1}{2\pi}\text{acos}\left(1 - \frac{\Psi[\dot{x}_b(n)]}{2\Psi[x_b(n)]}\right) \tag{9}$$

$$|a_b(n)| = \frac{\Psi[x_b(n)]}{1 - \left(1 - \frac{\Psi[\dot{x}_b(n)]}{2\Psi[x_b(n)]}\right)^2} \tag{10}$$

$$f_b(n) = \frac{1}{4\pi}\text{acos}\left(1 - \frac{\Psi[x_b(n+1) - x_b(n-1)]}{2\Psi[x_b(n)]}\right) \tag{11}$$

$$|a_b(n)| = \frac{2\Psi[x_b(n)]}{\sqrt{\Psi[x_b(n+1) - x_b(n-1)]}} \tag{12}$$

3.1.4 Band Estimation

The band estimator transforms DESA's output into time-windowed estimates with confidence bandwidths of dominant frequency (Eq. (13), Eq. (14)) and mean frequency (Eq. (15), Eq. (16)) (Potamianos and Maragos (1995), Shokouhi and Hansen (2017)).

$$F_b(t) = \frac{1}{N_t}\sum_{n=0}^{N_t-1} f_b(n) \tag{13}$$

$$W_b(t) = \frac{1}{N_t}\sum_{n=0}^{N_t-1} (f_b(n) - F_b(t))^2 \tag{14}$$

$$F_b(t) = \frac{\sum_{n=0}^{N_t-1} f_b(n)a_b^2(n)}{\sum_{n=0}^{N_t-1} a_b^2(n)} \tag{15}$$

$$W_b(t) = \sqrt{\frac{\sum_{n=0}^{N_t-1}(\frac{\dot{a}_b(n)}{2\pi})^2 + (f_b(n) - F_b(t))^2 a_b^2(n)}{\sum_{n=0}^{N_t-1} a_b^2(n)}} \tag{16}$$

The estimator selects amplitudes matching frequency estimates with confidence bandwidths tighter than some band-specific threshold $\digamma_b$, i.e.,

$$S_{pyk}(t,b) = \begin{cases} A(t,b), & W_b(t) < \digamma_b \\ 0, & \text{otherwise} \end{cases} \tag{17}$$

$$A(t,b) \triangleq \frac{1}{N_t}\sum_{n=0}^{N_t-1} a_b^2(n) \tag{18}$$

Potamianos and Maragos (1995) coined the term "pyknogram" for this band-pass presentation of frame energy. Phonotomizer normalizes each $A(t,b)$, dividing it by the total pyknogram frame energy. By default, $\digamma_b$ is pegged to the bandwidth w_b derived from Eqs. 1, 2, and 3 and the average bandwidth $\bar{w} = r/2B$ according to Eq. 19:

$$\digamma = 10^{\log_2(w_b) - 9\frac{w_b}{\bar{w}}} \tag{19}$$

This non-linear relationship was derived from the observation of consistent pre-visualization across band counts when progressively dividing $\digamma$ by 10 as the average bandwidth was halved due to doubling band count.

To guard against spurious formant detection, a noise floor parameter ν can be specified by the user. During comparison, the parameter ν is squared to match the squared amplitude estimate of the pyknogram and multiplied by the ratio of max band gain to total frame energy as an adaptive scaling factor to apply the same ν universally to noise and speech regions. Eq. 20 expresses the noise floor inequality used to filter unwanted bands.

$$A(t,b) > \nu^2 \frac{\max_{b'=0}^{B-1} A(t,b')}{\sum_{b'=0}^{B-1} A(t,b')} \tag{20}$$

This works equally well with voiced and unvoiced sounds while ignoring low-energy areas of disinterest. By default, $\nu = 0$, i.e., no noise floor is activated, but $\nu = 0.05$, i.e., a max floor of $1/20$ of the possible 32-bit float amplitude range, appears to work well in practice across band counts.

Harmonic suppression checks each band for integral factors of its own estimate among the lower bands, subtracting the energy of each detected coincident harmonic scaled by the reciprocal of the harmonic number (Daniel et al., 2024). Bands suppressed below the noise floor are dropped.

The OZGF frequency response sometimes causes estimates from neighboring bands to coalesce in one band, violating their respective boundaries. Any estimate that falls outside of a band's boundaries is likewise dropped.

3.1.5 Segmentation/Classification

The segmenter/classifier decides at each frame whether to update or segment a pending phone cluster based on band one-hot similarity and detected formants. An update to the pending cluster occurs if all of the following conditions are met. (1) Hot bands exist, and at least some are continuous. (2) The ratio of continuous to hot bands stays above a threshold. (3) The average formant estimate difference across continuous bands stays within confidence radii. Failure to meet these conditions triggers a formant similarity test between the pending cluster and the current frame. If this fails, segmentation occurs.

From the provided pyknogram, the segmenter selects the first four formant frequencies above 80 Hz and corresponding bandwidth estimates identified by Eq. 13 and Eq. 14, respectively, with non-zero pyknograms per Eq. 17. Regions of dense spectral energy may average band estimates, weighted by pyknogram. The values for each identified formant are converted into absolute values in Hz.

Rounding the frequency values to 16-bit signed integers from 32-bit floats gives a comfortable Nyquist ceiling of 32 768 Hz and facilitates a binary radix representation for the cluster labels. Per Fig. 2, interleaving the bits of each formant frequency (F_1–F_4) into a 64-bit unsigned integer yields a meaningful cluster label. Storing the clusters in memory in a PATRICIA trie (Morrison, 1968; Okasaki and Gill, 1998) keyed by label enables fast prefix searching where more definitive formants (e.g., F_1, F_2) guide initial branching and subtrees define formant cluster families.

Welford's (1962) online algorithm tracks each cluster's contributing frame count N_y, formant means vector $\boldsymbol{\mu}_y$ per Eq. (13), and confidence bandwidths vector $\boldsymbol{\rho}_y$ per Eq. (14). With frequencies as cluster means and confidences as standard deviations, cluster statistics can be merged per Chan et al. (1982) to form a new cluster in order to keep the cluster count $|Y|$ within reason. A merged cluster label is recalculated to match the potentially shifted mean formant vector.

Each new cluster z joins an existing cluster y if all mean formants overlap within respective confidences. Otherwise, the cluster z is added to the in-memory map of clusters, keyed by its label.

Areas of silence receive the apt label `0x0000000000000000` since no formants would have been detected. With learning frozen, indeterminate clusters receive the special label `0xffffffffffffffff`.

3.1.6 Model Persistence

Clusters are stored to disk in a packed binary format at the end of a training run. The file signature allocates 10 bytes for the extension `.phonotype` and 2 bytes each for Phonotomizer major, minor, and patch numbers for version disambiguation. A 64-bit unsigned integer header size indicates how far to seek to reach the model payload. No file header exists at present, so the value is zero (0).

Each persisted cluster stores a 64-bit integer label, a 64-bit frame count, and the four 16-bit formant frequencies and confidence bandwidths. Total cost of storage per cluster is therefore 32 bytes.

Since the clusters are uniquely identifiable and keyed by their labels, no additional metadata is stored. Model dumps write the file signature followed by each cluster's label and packed struct. Model loads check the file signature and read each label and packed struct, inserting them into the cluster map, until reaching the end of the file. Fig. 3 illustrates the cluster layout.

4 Experimental Setting

Table 1 describes the environment used for comparing Montreal Forced Aligner (MFA) v3.2.0 and ARTIC/Phonotomizer (A/P) v3.0.6.

The Mozilla Common Voice (MozCV) project (version: cv-corpus-12.0-2022-12-07) supplied training and test data in Irish and Twi (The Mozilla Foundation, 2022). Mozilla's designations "train" ("A") and "other" ("B") served as hold-out set partitions for Twi and cross-lingual tests, but only Irish

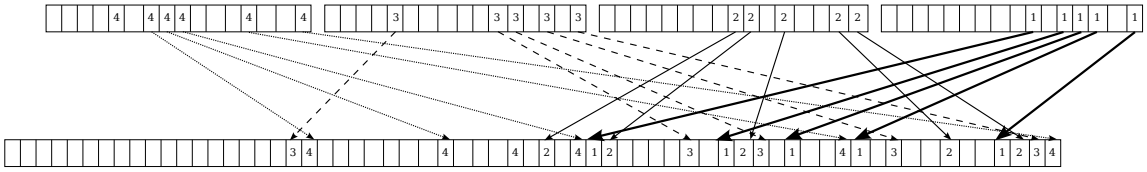

Figure 2: Phonotomizer Cluster Label Bit Interleaving. In this example, F_1 = 93 Hz, F_2 = 211 Hz, F_3 = 2101 Hz, and F_4 = 2953 Hz to yield the label `0x30115c2e9a4f`. More properly, these correspond to true $F_0 - F_3$ of /i/ in the Twi word "nim". A discontinuity in spectral energy between the fundamental frequency F_0 = 93 and true F_1 = 211 led to the shift in identification. The variance of F_0 among speakers suggests that future research should focus on identifying it apart from the other formants, perhaps using it as a basis for harmonic ratio clustering.

Header	Payload								
y	μ_{F1}	ρ_{F1}	μ_{F2}	ρ_{F2}	μ_{F3}	ρ_{F3}	μ_{F4}	ρ_{F4}	N_y
(8)	(2)	(2)	(2)	(2)	(2)	(2)	(2)	(2)	(8)

Figure 3: Phonotomizer Cluster Persistence Format. The size of the model grows as a product of cluster count $|Y|$. Bytes per field are given in parentheses.

Parameter	Value
OS	Ubuntu 24.04
CPU	Intel® Core™ i9-10900X, 3.70 GHz
Cores	10 cores × 2 threads
Memory	128 GB
Build System	GNU `make` + `g++` 13.3.0

Table 1: Experimental System Specifications

A was tested due to the high time cost of manual transcription. The Twi corpus received complete coverage, but Irish only received 15 transcriptions

Though spoken by many, these languages lack phonetic annotations and acoustic models in the literature. Several projects (e.g., WikiMedia Foundation (2008), Priva et al. (2021)) have abandoned efforts in Irish and Twi citing difficulties. These datasets thus support an unbiased, end-to-end evaluation of STT systems including pre-processing.

For Irish, the GNU `aspell` (GNU Project, 2020) *Gaeilge* word list was converted to International Phonetic Alphabet (IPA) symbols using rules defined by Ó Siadhail (1996) for the MFA pronunciation dictionary. For Twi, the MozCV transcriptions and a lexicon by Beermann et al. (2020) supplied words to convert to IPA using mappings from Zabolotskikh (2018) and Ager (2023).

Gold labels for post-hoc evaluation were manually transcribed with Praat as TextGrids (Boersma and Weenink, 2023a). TextGrid metrics calculations and Photonomizer classification output conversions used the PraatIO library (Mahrt, 2016).

Table 2 outlines the experimental parameters. DESA-1 seemed to perform best in results reported by Yantosca (2019), as well as on earlier, unpublished Phonotomizer experiments on the English LibriSpeech corpus (Panayotov et al., 2014). All band spacing options were tested, along with 4 different band counts. Fixing d_T at 10 ms and r at 16 kHz offered sufficient resolution to adequately identify formants and capture phone boundaries.

MFA trained for 35-40 epochs. Phonotomizer trained for 1 epoch except for zero shots. To establish a segmentation baseline apart from timing constraints, the underlying ARTIC pipeline applied no exactness modulation nor deadline adaptation.

Parameter	Value
Band spacing	Linear, Log, Quad
Band count (B)	40, 80, 160, 320
Filter order (O)	4
DESA Algorithm	DESA-1
Confidence threshold (F_b)	implicitly derived
One-hot similarity threshold	0.5
Noise floor (ν)	0.05
Sample Rate (r)	16 kHz
Data frame size (N_t)	160 samples (10 ms)
Training Sets	Twi A, Twi B, Irish A, None
Test Sets	Twi A, Twi B, Irish A

Table 2: Phonotomizer Evaluation Parameters

Given recent advances with transformers in phoneme classification (Baevski et al., 2020; Xu et al., 2021; Prat et al., 2024; Poli et al., 2024), we also attempted to establish baselines for these models. However, `PhonHuBERT` (Prat et al., 2024) would not build due to dependency conflicts even after trying multiple Python versions and pointing to standard packages instead of files local to the authors[1]. The restriction to TIMIT phonemes by `spokenlm-phoneme` (Poli et al., 2024) would not suffice to model Irish and Twi phones unrep-

[1] e.g., `certifi @ file:///...`

resented in English. Consequently, we chose to test Facebook's `wav2vec2-xlsr-53-espeak-cv-ft` model (Xu et al., 2021) derived from XLSR-53 (Conneau et al., 2021) for its adequate documentation and multilingual support.

5 Results

5.1 Model Size

Phonotomizer's unzipped acoustic model footprint ran about 10% of the size of MFA's zipped acoustic model format per Table 3. Standard deviation among tested Phonotomizer model sizes is given as a plus/minus adjustment.

Dataset	Clips	MFA 3.2.0	A/P 3.06
Twi A	12	160.0	19.0 $\pm$ 5.2
Twi B	217	2 596.0	243.3 $\pm$ 91.1
Irish A	537	6 244.0	512.3 $\pm$ 206.3

Table 3: Acoustic Model Footprints (KB)

5.2 Alignment Completion

MFA failed to generate alignments in some cases. Of the 537 clips in the Irish A set, MFA only generated 359 alignments, missing 5 from the set of 15 gold transcriptions. Phonotomizer generated alignments for all clips in all tests.

5.3 Clustering Metrics

Segmentation performance was evaluated with common clustering metrics having values in the range [0,1] in order to provide comprehensive and comparable statistics across the corpora tested.

Completeness is defined as $1 - \frac{H(Y|\Phi)}{H(Y)}$. $H(Y|\Phi)$ is cluster entropy conditioned by true labels, and $H(Y)$ is unconditional. It diagnoses oversegmentation, measuring how much each true label corresponds to a unique cluster (Wu et al., 2021).

Homogeneity is defined as $1 - \frac{H(\Phi|Y)}{H(\Phi)}$. $H(\Phi|Y)$ is true label entropy conditioned by clusters, and $H(\Phi)$ is unconditional. It diagnoses undersegmentation, measuring how much each cluster corresponds to the same true label (Wu et al., 2021).

Normalized Mutual Information (NMI) captures the mutual dependence between discovered clusters (Y) and true phonemes (Φ). Each cluster/phoneme joint probability mass function (PMF) is multiplied by the logarithm of that joint PMF over the product of the marginal PMFs, i.e., $MI = \sum_{y \in Y} \sum_{\phi \in \Phi} P_{Y,\Phi}(y, \phi) \log\left(\frac{P_{Y,\Phi}(y,\phi)}{P_Y(y) P_\Phi(\phi)}\right)$. This sum is then divided by the halved sum of the cluster and phoneme entropies for normalization, i.e., $NMI = 2\frac{MI}{H(Y)+H(\Phi)}$ (Wang et al., 2015).

Fig. 4 demonstrates that MFA generally fared better at completeness and Phonotomizer at homogeneity, indicating tendencies to under- and oversegmentation, respectively. Phonotomizer zero-shot runs scored highest overall at NMI, suggesting the best balance. Peak performance appears to coincide with $B = 80$ bands, i.e., an average bandwidth of 100 Hz. The low completeness scores suggest room for improvement on the classifier.

MFA performed best with the richest datasets but floundered for lack of data, especially on cross-lingual tests. Data scarcity impacted Phonotomizer less, though trained model runs exhibited evidence of overfitting to the respective training sets. Despite this, Phonotomizer often outperformed MFA when tested on a language other than its training set, lending credence to the universality of its *phonetic* as opposed to MFA's *phonemic* approach.

A salient pyknogram example in Fig. 5 offers insight into Phonotomizer's strengths and weaknesses. Segment boundaries correspond well with clear changes in spectral energy, though oversegmented transitions between phones illustrate the issues with completeness. Conflation of neighboring sounds can occur, e.g., in the presence of high-energy, turbulent sounds like the sibilant /s/ that bleed into adjacent phones. However, Phonotomizer's automatic noise filtration guards against spurious identification of silence as a region of interest. In this depicted clip, which runs for 2.7 s, low-energy noise at the start and end confuse MFA and exacerbate the effects of its text dependency, as shown in Fig. 6.

5.4 Wav2Vec2Phoneme Comparison

Facebook's `wav2vec2-xlsr-53-espeak-cv-ft` model, a.k.a. Wav2Vec2Phoneme (W2V2P), produced plausible phone sequences in zero-shot testing against the MozCV Twi corpus. Unlike MFA, it lacked precise timing for producing TextGrids, conflating silence and model indecision and tagging each phone on one 20 ms frame near the phone's end. Without delimiter tags, the pad symbol overloading made initial boundaries indeterminable.

Table 4 depicts confusion tables for the top 10 test matches against 3 gold phones: /b/, /ɔ/, and /ɕ/. Each test phone's time percentage per selected gold phone over a complete run on the Twi corpus appears in columns for W2V2P with standard pretraining and untrained ARTIC/Phonotomizer (A/P)

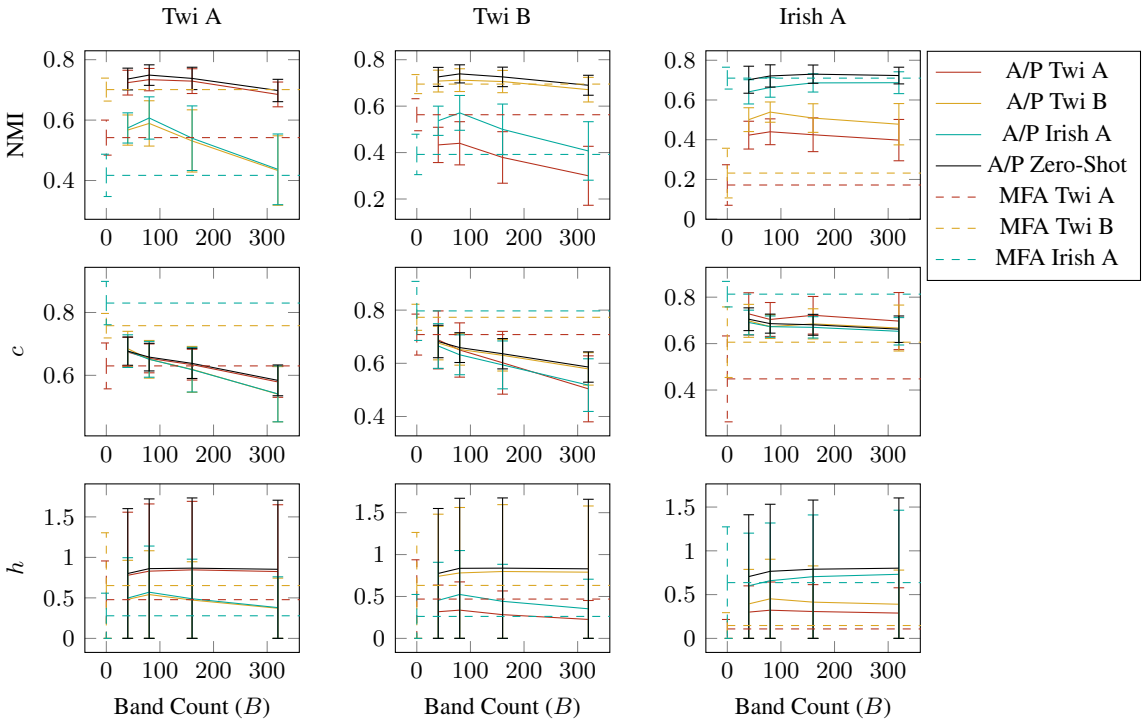

Figure 4: Average model clustering performance: Normalized Mutual Information (NMI), Completeness (c), and Homogeneity (h). Errors bars show standard deviation. Poor $c \rightarrow$ oversegmentation. Poor $h \rightarrow$ undersegmentation. NMI gives a balanced, holistic clustering score. Column titles show test set. Trend lines indicate training set if any.

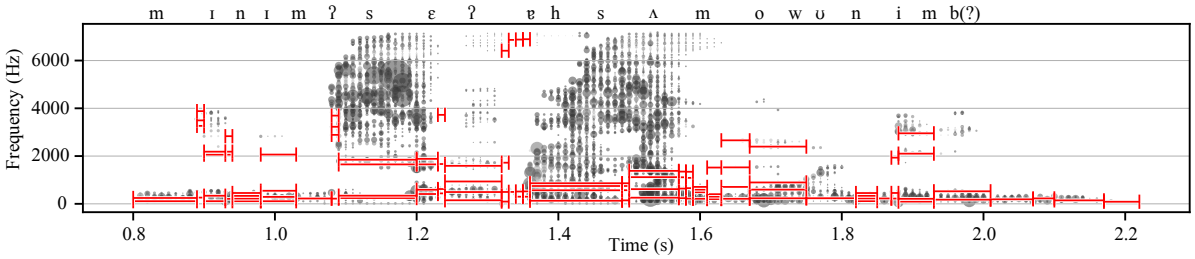

Figure 5: Pyknogram of Twi utterance “minim sɛ asamoa nim” (English: “I know that heaven knows.”) in grayscale. Band count = 160, spacing = logarithmic, zero-shot. Formant selections at segmentation intervals in red. Voiced sounds (e.g., /mɪnɪm/, /nim(b?)/) segment well, albeit with some oversegmentation during transitions between phones due to feature instability. Sibilants tend to bleed (e.g., /sɛ/, /ɐhsʌ/).

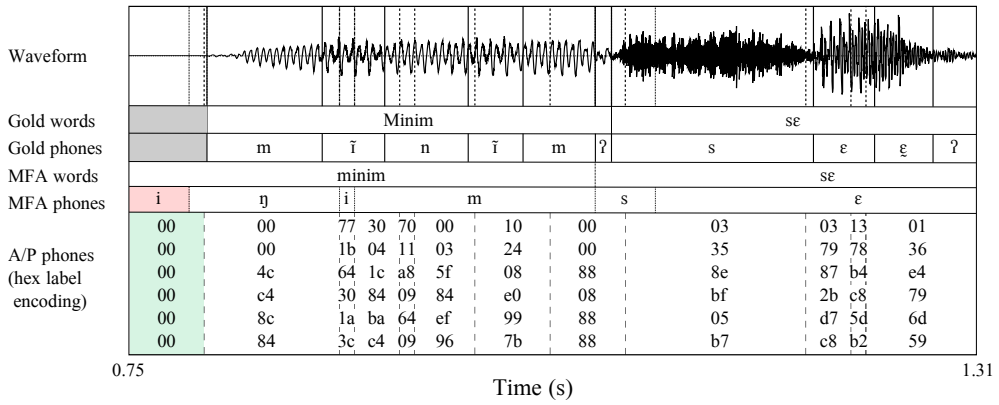

Figure 6: Sample Twi TextGrid with MFA Word Misalignment. The MFA model *trained on this clip* hallucinates the majority of the first word in the initial silence and covers the true /nɪm/ with a single /m/ classification. ARTIC/Phonotomizer (A/P) *zero-shot* correctly identifies the initial silence with segments closer to ground truth.

with online training enabled for each test clip alone.

XLSR-53's training languages share the first two phones. Owing to its phonemic importance, Twi grants /ɔ/ its own letter "ɔ". /ɕ/ is common in Twi but unrepresented in the XLSR-53 training set.

Due to its prevalence by frame, padding dominates the W2V2P matches for each gold phone by time covered. The next best actual phone matches prove correct or close substitutes given training.

However, matching /ɔ/ to /o/, /ɔ/, and /oː/ hints at systemic issues also seen in similar phone pairs like /ɛ/ and /e/. Overall, W2V2P's labels straddle multiple phonetic encodings, producing not only IPA but also notation like "i.5" and "ong5"[2], possibly conflating homographic symbols from different contexts. Two phonemizers sourced the phonemic transcriptions of the 3 training sets from lexemic text conversions instead of vocal analysis (Xu et al., 2021)[3]. Ignoring the true phonetic context risks undervaluing uncommon accents and phones.

A/P's confusion results are grouped by prefix into hyperclusters due to frequent hapax, i.e., one-off, production. Truncating 7 hexadecimal digits from the right functionally truncates 7 binary digits per formant, yielding a uniform radius per formant of about 127 Hz for each hypercluster in 4 dimensions. Being all zeroes in all clusters, the 2 most significant digits are elided for legibility. Improvements in the formant picking algorithm should generate better clusters with tighter grouping since the system exhibited a bias toward spurious identification of formant peaks at lower frequencies.

Despite this, the A/P results for /b/ favor the top two hyperclusters with over 50% coverage, and /ɔ/ and /b/ demonstrate strong prefix similiarity. The spread for /ɕ/ is more uniform as might be expected for a turbulent sibilant with less defined formants. None of the depicted hyperclusters show conflation with silence (i.e., 00 00 00 0).

Gold	W2V2P	t (%)	A/P	t (%)
b		90.97	00 00 00 8	43.06
b	b	6.61	00 00 40 8	11.34
b	o	0.48	00 00 04 c	2.97
b	a	0.45	00 00 08 0	2.44
b	u	0.31	00 00 48 0	2.12
b	m	0.24	00 00 40 c	2.03
b	d	0.16	00 00 44 c	1.16
b	v	0.16	00 00 0c 4	1.05
b	i	0.16	00 00 04 0	0.88
b	p	0.10	00 02 40 8	0.88
ɔ		72.39	00 00 48 c	10.51
ɔ	o	12.67	00 00 40 8	4.78
ɔ	a	3.39	00 00 48 0	3.39
ɔ	w	1.69	00 00 48 8	2.22
ɔ	u	1.59	00 00 08 8	2.05
ɔ	b	1.39	00 00 00 8	1.93
ɔ	ɔ	1.29	00 00 40 c	1.81
ɔ	oː	0.90	00 00 48 4	1.66
ɔ	n	0.80	00 00 6a 2	1.66
ɔ	h	0.68	00 00 c0 4	1.55
ɕ		83.80	00 34 5e 1	2.57
ɕ	ʃ	7.90	01 e6 a4 3	2.15
ɕ	s	5.19	07 89 a0 4	1.91
ɕ	t	0.55	00 30 58 5	1.79
ɕ	e	0.51	01 e6 b4 7	1.52
ɕ	f	0.51	00 73 57 f	1.50
ɕ	i	0.49	00 79 ad 6	1.50
ɕ	ts	0.27	01 e6 a0 5	1.50
ɕ	ɪ	0.25	00 73 57 6	1.49
ɕ	n	0.18	00 f3 57 a	1.49

Table 4: Top-10 Twi Confusion Tables for /b/, /ɔ/, /ɕ/ A/P prefix similarities are highlighted in color. A/P used logarithmic banding ($B = 80$).

6 Conclusions

Phonotomizer proves the possibility of making more with less with its zero-shot runs consistently outscoring MFA on NMI. Spot evaluations of the generated TextGrids confirmed MFA's noise intolerance in contrast to Phonotomizer's rugged adaptability. The pyknograms generated by Phonotomizer as inputs for segmentation and classification visually confirmed the efficacy of the methodology and significantly accelerated development.

However, Phonotomizer's models, while more compact, require improvement on the classification end. We are exploring improvements to the formant picker by dividing the spectrum for more granulated comparison of energy regions across time. We are also looking to reformulate the cluster format to better handle speaker variability, e.g., rebasing the formant definitions as harmonic ratios of the fundamental frequency. Finding a path toward conversion to IPA and evaluating electricity consumption constitute open areas of research.

In this vein, porting the framework to an embedded context could prove useful to linguists in the field working under sparse network connectivity and severe resource constraints. Additionally, the small footprint of Phonotomizer and ARTIC lends itself to privacy-sensitive, low-impact approaches to NLP in diametric opposition to the titanic quantities of e-waste generated by the state of the art. This could prove a boon to endangered language communities seeking to preserve their cultural heritage in the midst of a growing homogenization of the spaces we inhabit, digital or otherwise.

[2]NB: not "ong5", but "ong5", i.e., g = \U+0261.

[3]cf. §3.2 of their paper.

7 Limitations

7.1 Gold Label Accuracy

The gold labels were transcribed by the primary author alone over the course of several months. As such, more recent transcriptions likely possess higher quality than earlier transcriptions, and there is some degree of *intra*-transcriber disagreement.

For instance, the Twi reference clip used during development was re-transcribed after the first pass of transcriptions was completed to better capture the phones involved and to distinguish nasals and creaky voice with diacritics. Editing the gold label dataset is an ongoing process, and revisions are tracked using `git` for source control. The `goldgrids-twi` repo used commit `a3bb57e5fb3574da02e8891efb2ba48c7b69bf52`, and the `goldgrids-gle` repo used commit `53568849d8db7ca332f8e892c0a6001eac6911a1`.

7.2 Applicability

While Phonotomizer has built-in facilities for noise tolerance, the current implementation assumes a single speaker. Source separation is not attempted but might prove useful in future development.

Phonotomizer does not correct or compensate for disfluencies or pathological speech conditions, e.g., dysarthria. While Phonotomizer's transparent capture of the raw phonetic data can support studies in speech pathology or field linguistics, end-to-end lexemic or orthographic transcription applications built on top of Phonotomizer would need to account for this limitation and adjust accordingly.

7.3 Experimental Parameters

The executable and supporting pipeline modules are written to the C++20 standard and have been built with `g++-13` and tested on Ubuntu 24.04 only. Audio ingestion and resampling depend on `ffmpeg` (FFmpeg Project, 2019) but use Praat's prelude skip count to throw away some initial samples (Boersma and Weenink, 2023b). Unfortunately, there is no standard behavior across MP3 encoders with respect to padding samples. Even ffmpeg has open issues on this behavior from version to version (Robertson, 2023). We chose to adhere to Praat's behavior for simplicity and coherence with manually transcribed ground truths.

8 Ethical Considerations

Anonymous contributors to The Mozilla Foundation (2022) may withdraw their data, but no such requests were received for Irish or Twi. In accord with the license, no diarization was attempted.

All code and models used for experiments were written and developed by the primary author with the following exceptions:

- GNU C++ standard library (Carlini et al., 2020), which was used throughout the project
- ffmpeg (FFmpeg Project, 2019), which was used for audio ingestion and resampling
- C++ biquad source code (Redmon, 2012), which the primary author adapted for the OZGF bank stage
- JSON for Modern C++ (Lohmann, 2022), which was used for parsing Phonotomizer pipeline schedules
- libcester (Azeez, 2020), which was used for unit tests in the ARTIC framework and Phonotomizer
- Montreal Forced Aligner (MFA) 3.2.0 (McAuliffe et al., 2017), which was used for comparison in experiments
- Facebook's Wav2Vec2Phoneme (Xu et al., 2021) Hugging Face model `wav2vec2-xlsr-53-espeak-cv-ft` based on XLSR-53 (Conneau et al., 2021), which was used for further experimental comparisons per reviewer feedback
- PraatIO (Mahrt, 2016), which was used to generate TextGrids

The development and experiments described in this paper did not utilize research products (code, models, etc.) implicated in or from companies under investigation for alleged abuse of workers (cf. Phillips, 2018; Perrigo, 2022; Perrigo, 2023; Bartholomew, 2023; Musinga et al., 2024). The tests of Facebook's `wav2vec2-xlsr-53-espeak-cv-ft` model may constitute the exception which proves this rule, but we deemed comparison with our gold transcriptions necessary to substantiate a robust critique of these resource-intensive methods. No large language models (LLMs) were employed in the writing or editing of this paper.

References

Simon Ager. 2023. Omniglot: Twi. Online article.

Adewale Azeez. 2020. libcester: a robust header only unit testing framework for c and c++ programming language. Software library.

Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. Wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in neural information processing systems*, 33:12449–12460.

Loïc Barrault, Yu-An Chung, Mariano Cora Meglioli, David Dale, Ning Dong, Paul-Ambroise Duquenne, Hady Elsahar, Hongyu Gong, Kevin Heffernan, John Hoffman, et al. 2023a. SeamlessM4T-massively multilingual & multimodal machine translation. *arXiv preprint arXiv:2308.11596*.

Loïc Barrault, Yu-An Chung, Mariano Coria Meglioli, David Dale, Ning Dong, Mark Duppenthaler, Paul-Ambroise Duquenne, Brian Ellis, Hady Elsahar, Justin Haaheim, et al. 2023b. Seamless: Multilingual expressive and streaming speech translation. *arXiv preprint arXiv:2312.05187*.

Jem Bartholomew. 2023. Q&A: Uncovering the labor exploitation that powers AI. *Columbia Journalism Review*.

Sanjoy Baruah, Alan Burns, Robert I Davis, and Yue Wu. 2023. Optimally ordering IDK classifiers subject to deadlines. *Real-Time Systems*, 59(1):1–34.

Dorothee Beermann, Lars Hellan, Pavel Mihaylov, and Anna Struck. 2020. Developing a Twi (Asante) dictionary from Akan interlinear glossed texts. In *Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020)*, pages 294–297. European Language Resources Association (ELRA).

Mathieu Bernard and Hadrien Titeux. 2021. Phonemizer: Text to phones transcription for multiple languages in python. *Journal of Open Source Software*, 6(68):3958.

Saurabhch Bhati, Herman Kamper, and K. Sri Rama Murty. 2018. Phoneme based embedded segmental k-means for unsupervised term discovery. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5169–5173.

Paul Boersma and David Weenink. 2023a. Praat: doing phonetics by computer. Software program.

Paul Boersma and David Weenink. 2023b. Praat mp3 decoder source. Software source code.

Paolo Carlini, Phil Edwards, Doug Gregor, Benjamin Kosnik, Dhruv Matani, Jason Merrill, Mark Mitchell, Nathan Myers, Felix Natter, Stefan Olsson, Johannes Singler, Ami Tavory, and Jonathan Wakely. 2020. *The GNU C++ Library*, 13.3.0 edition. Free Software Foundation.

Tony F. Chan, Gene H. Golub, and Randall J. LeVeque. 1982. Updating formulae and a pairwise algorithm for computing sample variances. In *COMPSTAT 1982 5th Symposium held at Toulouse 1982: Part I: Proceedings in Computational Statistics*, pages 30–41. Springer.

Yi-Chen Chen, Sung-Feng Huang, Hung-yi Lee, Yu-Hsuan Wang, and Chia-Hao Shen. 2019. Audio word2vec: Sequence-to-sequence autoencoding for unsupervised learning of audio segmentation and representation. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 27(9):1481–1493.

Alexis Conneau, Alexei Baevski, Ronan Collobert, Abdelrahman Mohamed, and Michael Auli. 2021. Unsupervised cross-lingual representation learning for speech recognition. *Interspeech 2021*.

Kamran Daniel, Lauri Kütt, Muhammad Naveed Iqbal, Noman Shabbir, Hadi Ashraf Raja, and Muhammad Usman Sardar. 2024. A review of harmonic detection, suppression, aggregation, and estimation techniques. *Applied Sciences*, 14(23):10966.

R. O. Duda, P. E. Hart, and D. G. Stork. 2001. *Pattern Classification*. John Wiley & Sons, Inc.

Yassine El Kheir, Hamdy Mubarak, Ahmed Ali, and Shammur Chowdhury. 2024. Beyond orthography: Automatic recovery of short vowels and dialectal sounds in Arabic. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13172–13184, Bangkok, Thailand. Association for Computational Linguistics.

FFmpeg Project. 2019. Ffmpeg. Software library.

Biao Fu, Kai Fan, Minpeng Liao, Yidong Chen, Xiaodong Shi, and Zhongqiang Huang. 2024. Wav2vec-S: Adapting pre-trained speech models for streaming. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 11465–11480, Bangkok, Thailand. Association for Computational Linguistics.

P. Gangamohan and Suryakanth V. Gangashetty. 2019. Epoch extraction from speech signals using temporal and spectral cues by exploiting harmonic structure of impulse-like excitations. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6505–6509. IEEE.

Wei Gao, Ahmad Hashemi-Sakhtsari, and Mark D. McDonnell. 2020. End-to-end phoneme recognition using models from semantic image segmentation. In *2020 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7.

J. Garofolo, Lori Lamel, W. Fisher, Jonathan Fiscus, D. Pallett, N. Dahlgren, and V. Zue. 1992. TIMIT acoustic-phonetic continuous speech corpus. *Linguistic Data Consortium*.

GNU Project. 2020. Available aspell dictionaries. Software library.

Hossein Hamooni and Abdullah Mueen. 2014. Dual-domain hierarchical classification of phonetic time series. In *2014 IEEE international conference on data mining*, pages 160–169. IEEE.

Awni Hannun, Carl Case, Jared Casper, Bryan Catanzaro, Greg Diamos, Erich Elsen, Ryan Prenger, Sanjeev Satheesh, Shubho Sengupta, Adam Coates, and Andrew Y. Ng. 2014. Deep speech: Scaling up end-to-end speech recognition. *arXiv preprint arXiv:1412.5567*.

Herman Kamper. 2019. Truly unsupervised acoustic word embeddings using weak top-down constraints in encoder-decoder models. In *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6535–3539. IEEE.

Herman Kamper, Aren Jansen, and Sharon Goldwater. 2016. Unsupervised word segmentation and lexicon discovery using acoustic word embeddings. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):669–679.

A. G. Katsiamis, E. M. Drakakis, and R. F. Lyon. 2009. A biomimetic 4.5 w, 120+ db, log-domain cochlea channel with AGC. *IEEE J. Solid-State Circuits*, 44(3):1006–1022.

Felix Kreuk, Yaniv Sheena, Joseph Keshet, and Yossi Adi. 2020. Phoneme boundary detection using learnable segmental features. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8089–8093.

Binghuai Lin and Liyuan Wang. 2022. Learning acoustic frame labeling for phoneme segmentation with regularized attention mechanism. In *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7882–7886.

Zoey Liu, Nitin Venkateswaran, Eric Le Ferrand, and Emily Prud'hommeaux. 2024. How important is a language model for low-resource ASR? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 206–213, Bangkok, Thailand. Association for Computational Linguistics.

Niels Lohmann. 2022. Json for modern c++. Software library.

Sasha Luccioni, Bruna Trevelin, and Margaret Mitchell. 2024. The environmental impacts of AI – policy primer. In *Hugging Face Blog*.

R. Lyon. 1996. The all-pole gammatone filter and auditory models. Technical report, Apple Computer, Inc.

Tim Mahrt. 2016. PraatIO. Software library.

P. Maragos, J. F. Kaiser, and T. F. Quatieri. 1993. Energy separation in signal modulations with application to speech analysis. 41(10):3024–3051.

Michael McAuliffe, Michaela Socolof, Sarah Mihuc, Michael Wagner, and Morgan Sonderegger. 2017. Montreal forced aligner: Trainable text-speech alignment using Kaldi. In *Proc. Interspeech 2017*, pages 498–502.

Donald R Morrison. 1968. Patriciapractical algorithm to retrieve information coded in alphanumeric. *Journal of the ACM (JACM)*, 15(4):514–534.

D.K. Musinga, Asike Makhandia, and Mativo J. 2024. Meta Platforms, Inc and 2 others v Motaung and 186 others; Kenya National Human Rights and Equality Commission and 14 others (interested parties) [2024] KECA 1262 (KLR). *Kenya Law Review*.

Anh-Vu Nguyen, Albert M. K. Cheng, and Thomas Carroll. 2024. Work-in-progress: Utilizing probabilistic analysis to fine-tune optimal IDK cascades. In *2024 IEEE Real-Time Systems Symposium (RTSS)*, pages 439–442.

Chris Okasaki and Andrew Gill. 1998. Fast mergeable integer maps. In *Workshop on ML*, pages 77–86.

Yasuaki Okuda, Ryo Ozaki, Soichiro Komura, and Tadahiro Taniguchi. 2022. Double articulation analyzer with prosody for unsupervised word and phone discovery. *IEEE Transactions on Cognitive and Developmental Systems*, 15(3):1335–1347.

V. Panayotov, D. Povey, and S. Khudanpur. 2014. Librispeech language models, vocabulary and g2p models. Linguistic corpus.

Billy Perrigo. 2022. Inside Facebook's African sweatshop. *Time*.

Billy Perrigo. 2023. Exclusive: OpenAI used kenyan workers on less than $2 per hour to make ChatGPT less toxic. *Time*.

Craig Phillips. 2018. Filmmakers navigate the secretive world of social media censorship in atmosphere of fear. *Independent Lens, PBS*.

Mark A Pitt, Keith Johnson, Elizabeth Hume, Scott Kiesling, and William Raymond. 2005. The buckeye corpus of conversational speech: Labeling conventions and a test of transcriber reliability. *Speech Communication*, 45(1):89–95.

Maxime Poli, Emmanuel Chemla, and Emmanuel Dupoux. 2024. Improving spoken language modeling with phoneme classification: A simple fine-tuning approach. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*.

A. Potamianos and P. Maragos. 1995. Speech formant frequency and bandwidth tracking using multiband energy demodulation. In *Proc. 1995 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, volume 1, pages 784–787.

Amaury Prat, Runxuan Yang, and Xiaolin Hu. 2024. Phonhubert: A phoneme transcription tool for song datasets. In *Advances in Neural Networks – ISNN 2024*, pages 123–132, Singapore. Springer Nature Singapore.

Uriel Cohen Priva, Emily Strand, Shiying Yang, William Mizgerd, Abigail Creighton, Justin Bai, Rebecca Mathew, Allison Shao, Jordan Schuster, and Daniela Wiepert. 2021. XPF. Linguistic corpus.

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

N. Redmon. 2012. Biquad c++ source code. Blog post.

Morgane Riviere, Armand Joulin, Pierre-Emmanuel Mazare, and Emmanuel Dupoux. 2020. Unsupervised pretraining transfers well across languages. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Garry Robertson. 2023. Change in audio track initial_padding behaviour from ffmpeg v5 to ffmpeg v6. Bug report.

Steffen Schneider, Alexei Baevski, Ronan Collobert, and Michael Auli. 2019. Wav2vec: Unsupervised pre-training for speech recognition. *arXiv preprint arXiv:1904.05862*.

N. Shokouhi and J. H. L. Hansen. 2017. Teager-Kaiser energy operators for overlapped speech detection. 25(5):1035–1047.

Yuki Tada, Yoshinobu Hagiwara, and Tadahiro Taniguchi. 2017. Comparative study of feature extraction methods for direct word discovery with npbdaa from natural speech signals. In *2017 Joint IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL-EpiRob)*, pages 164–171. IEEE.

Chihiro Taguchi and David Chiang. 2024. Language complexity and speech recognition accuracy: Orthographic complexity hurts, phonological complexity doesn't. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15493–15503, Bangkok, Thailand. Association for Computational Linguistics.

Akira Taniguchi, Hiroaki Murakami, Ryo Ozaki, and Tadahiro Taniguchi. 2023. Unsupervised multimodal word discovery based on double articulation analysis with co-occurrence cues. *IEEE Transactions on Cognitive and Developmental Systems*.

Tadahiro Taniguchi, Shogo Nagasaka, and Ryo Nakashima. 2016. Nonparametric bayesian double articulation analyzer for direct language acquisition from continuous speech signals. *IEEE Transactions on Cognitive and Developmental Systems*, 8(3):171–185.

The Mozilla Foundation. 2022. Mozilla common voice. Annotated datasets.

Taira Tsuchiya, Naohiro Tawara, Testuji Ogawa, and Tetsunori Kobayashi. 2018. Speaker invariant feature extraction for zero-resource languages with adversarial learning. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2381–2385. IEEE.

A. Vaswani. 2017. Attention is all you need. *Advances in Neural Information Processing Systems*.

Haipeng Wang, Tan Lee, Cheung-Chi Leung, Bin Ma, and Haizhou Li. 2015. Acoustic segment modeling with spectral clustering methods. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23(2):264–277.

B. P. Welford. 1962. Note on a method for calculating corrected sums of squares and products. *Technometrics*, 4(3):419–420.

WikiMedia Foundation. 2008. Proposals for closing projects/closure of Twi wiktionary.

Bin Wu, Sakriani Sakti, Jinsong Zhang, and Satoshi Nakamura. 2021. Tackling perception bias in unsupervised phoneme discovery using DPGMM-RNN hybrid model and functional load. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 29:348–362.

Qiantong Xu, Alexei Baevski, and Michael Auli. 2021. Simple and effective zero-shot cross-lingual phoneme recognition. *arXiv preprint arXiv:2109.11680*.

Michael S. Yantosca. 2019. ARTIC: An adaptive realtime imprecise computation pipeline for audio analysis. MS thesis, University of Houston.

Yang Yu, Jiahui Wang, Yu Liu, Pingfeng Yu, Dongsheng Wang, Ping Zheng, and Meng Zhang. 2024. Revisit the environmental impact of artificial intelligence: the overlooked carbon emission source? *Frontiers of Environmental Science & Engineering*, 18(12):1–5.

Polina Zabolotskikh. 2018. Introductory materials: Akan Twi Asante. Online article.

Míchéal Ó Siadhail. 1996. *Learning Irish*, 3 edition. Yale University Press.