Living Speech: Script-Native Automatic Speech Recognition for N'Ko

Mohamed Diomande
Independent Researcher
contact@mohameddiomande.com

Abstract

Every published Bambara automatic speech recognition system produces Latin-script output. For the 40+ million N'Ko-literate speakers across West Africa, the entire ASR field has been writing in a foreign script. We build the first audio-to-N'Ko ASR system, converting Bambara speech directly to N'Ko script without Latin as an intermediary.

Our approach exploits a structural advantage. N'Ko is a bijective phoneme-to-grapheme script: every Manding phoneme maps to exactly one Unicode character, tone is marked explicitly, and there are no spelling irregularities. This bijectivity reduces the CTC decoder's output space to 66 classes (65 N'Ko codepoints plus blank), compared to the effectively larger combinatorial space of Latin Bambara (26 base letters plus digraphs, tone-unmarked).

We present a four-version architecture progression. **V1**: A BiLSTM CTC decoder (5.4M parameters) on frozen Whisper large-v3 features achieves 56% CER. A **28-configuration architecture search** over BiLSTM, Transformer, and Conformer variants identifies the optimal design. **V3**: A Transformer CTC decoder (46.9M parameters) with 6 layers, 768 hidden dimension, and 4× downsampling achieves 33% CER and 70% WER. **V4**: Whisper LoRA fine-tuning (rank=32, layers 24–31, 5.9M trainable parameters) on A100 reduces validation loss from 0.884 to 0.290 over 30 epochs, improving WER by 11% over V3 and boosting prediction confidence from 0.46 to 0.82 (79% improvement). Per-sample evaluation shows LoRA wins on 20/50 samples, base wins on 17/50, and 13/50 tie, with dramatic improvements on worst-case inputs (sample au30: WER 15.8 → 1.0).

A deterministic cross-script bridge with 6 documented bug classes converts Latin Bambara transcriptions to N'Ko training targets. A 4-state finite-state machine encoding N'Ko syllable phonotactics guarantees 100% structural validity at negligible runtime cost. A downstream translation pipeline using NLLB-200 fine-tuned on 8,640 pairs across 4 language directions achieves real-time N'Ko-to-English/French translation at 67ms per sentence.

Total compute cost: $14.

1 Introduction

In 1949, Solomana Kanté designed N'Ko as a writing system for the Manding languages of West Africa. The script has 27 base characters occupying Unicode block U+07C0--U+07FF (standardized 2006), writes right-to-left, and was engineered with a property that no evolved writing system possesses: a strict bijection between phonemes and graphemes. Every sound in Manding has exactly one character. Every character represents exactly one sound. Tone is marked explicitly with combining diacritics. There are no irregular spellings, no silent letters, no digraphs, no context-dependent pronunciation rules. The name "N'Ko" means "I say" in all Manding languages—Bambara, Maninka, Dioula, and their varieties.

The paradox at the center of this paper is that N'Ko is arguably the best-designed writing system for automatic speech recognition, and no ASR system has ever targeted it. The current state of the art for Bambara ASR—MALIBA-AI bambara-asr-v3 at 45.73% WER (MALIBA-AI, 2024), sudoping01/bambara-asr-v2 at 25.07% WER on its test split—produces Latin-script output using orthographic conventions designed by French colonial linguists in the 20th century. These conventions use digraphs ("ny" for a single nasal palatal phoneme, "ng" for a velar nasal), omit tone marking, and include irregular spellings inherited from French phonological conventions.

For a child in Kankan, Guinea, who speaks Maninka and reads N'Ko, every existing ASR

system writes in the wrong alphabet. This is not a transliteration problem that can be solved with a post-processing script. The Latin orthography actively conceals phonemic information—tone, nasalization, vowel quality—that N'Ko was designed to express. Recovering this information from Latin output requires knowledge that the ASR system discarded upstream.

We build the first audio-to-N'Ko ASR system. The system converts Bambara speech directly to N'Ko script through a pipeline that never routes through Latin as an intermediate representation at inference time. We exploit N'Ko's bijectivity as a computational advantage: the CTC decoder's output space is structurally simpler, more phonemically transparent, and more amenable to hard-constraint post-processing than any Latin-based alternative.

This paper makes nine contributions:

1. The **phonetic transparency hypothesis**: a formal analysis of why N'Ko's bijective phoneme-grapheme mapping reduces CTC output space complexity relative to Latin Bambara (§3).

2. A **deterministic cross-script bridge** from Latin Bambara transcriptions to N'Ko, with 6 documented bug classes that catalogue exactly where colonial orthography obscures phonemic information (§4).

3. A **28-configuration architecture search** over BiLSTM, Transformer, and Conformer variants with three hidden dimensions, three depths, and three downsampling rates (§5.3).

4. **Four architecture versions** (V1 through V4) with complete training progression (§5.2–§5.6).

5. **V4 Whisper LoRA results**: 30 epochs on A100, validation loss 0.884 → 0.290, with per-sample evaluation showing LoRA wins on 40% of samples with dramatic worst-case improvements (§5.6).

6. A **4-state finite-state machine** encoding N'Ko syllable phonotactics as hard constraints on CTC output, with formal specification and validation statistics (§6).

7. A **cross-script translation pipeline**: N'Ko → Latin → NLLB-200 → English/French, with NLLB-200 fine-tuned on 8,640 pairs achieving real-time inference (§7).

8. **Distributed inference architecture**: pipeline parallelism over Thunderbolt 5 at 0.4ms inter-node latency (§8).

9. Evidence that **self-attention enables circuit formation that BiLSTM cannot**, and that the cross-script bridge **recovers information that colonial orthography encoded away** (§9).

2 Related Work

2.1 Bambara and Manding ASR

The current state of the art for Bambara ASR is MALIBA-AI bambara-asr-v3, a LoRA fine-tune of Whisper large-v3 achieving 45.73% WER on the MALIBA-AI benchmark corpus and 13.23% WER under normalized evaluation (MALIBA-AI, 2024). The sudoping01/bambara-asr-v2 model achieves 25.07% WER on its test split using a different data partition. Neither system produces N'Ko output. FarmRadioInternational/bambara-whisper-asr is publicly available (ungated) and serves as the transcription backend in our data pipeline.

The first Bambara LLM, sudoping01/maliba-llm (Gemma-3n fine-tuned on 1M examples), was released in 2026 and supports Bambara-French-English code-switching in Latin script. A 2026 survey of Bambara ASR (Bambara ASR Survey, 2026) catalogues 11 publicly available models, all targeting Latin-script output.

The RobotsMali/afvoices dataset (612 hours) and bam-asr-early (37 hours, CC-BY-4.0) (RobotsMali, 2024) are the primary public Bambara speech corpora. The Bayelemabaga corpus (Coulibaly et al., 2025) provides 46,976 Bambara-French parallel segments. The WMT 2023 N'Ko shared task (Barrault et al., 2023) established NMT baselines for N'Ko script (30.83 chrF++ en→nko on FLoRes-devtest) using 130,850 parallel segments from the nicolingua collection.

To our knowledge, no prior work targets N'Ko as the output script for ASR, making our system the first of its kind.

2.2 Low-Resource ASR

The standard recipe for low-resource ASR is transfer learning from large pre-trained acoustic mod-

els: Whisper (Radford et al., 2023), wav2vec 2.0 (Baevski et al., 2020), and HuBERT (Hsu et al., 2021). These approaches reduce data requirements substantially but remain constrained by target script structure: Latin digraphs, irregular spellings, and unmarked tone add decoder complexity that is entirely unnecessary for a 1:1 phoneme-grapheme script.

CTC (Connectionist Temporal Classification) was introduced by Graves et al. (2006) as a method for labeling unsegmented sequences without explicit alignment. CTC's output vocabulary size is linear in the output projection parameter count; smaller, more structured output alphabets directly reduce decoder parameter requirements. This structural economy is the central computational advantage we exploit.

SpecAugment (Park et al., 2019)—time and frequency masking of mel spectrograms—provides the primary data augmentation strategy for low-resource ASR and is used in our V3 and V4 architectures.

Whisper large-v3 (Radford et al., 2023), trained on 680,000 hours of multilingual audio, serves as our acoustic encoder (frozen in V1–V3, partially unfrozen via LoRA in V4). Frozen encoder feature extraction has been validated in several low-resource settings as a practical alternative to full fine-tuning when labeled target-language data is scarce (San et al., 2021).

2.3 Script-Specific ASR

The relationship between target script structure and ASR difficulty has been studied in several language families. Tonja et al. (2023) note that Ethiopic/Ge'ez syllabary structure provides natural alignment between acoustic syllables and written characters, reducing CTC decoder complexity. Kakwani et al. (2020) found that Devanagari's largely phonemic orthography produces better ASR training efficiency than English's highly non-phonemic spelling.

Our work contributes to this literature by providing the first controlled experiment where the *same* speech (Bambara audio) is decoded to two different scripts (Latin and N'Ko), isolating the effect of target script structure on ASR accuracy.

2.4 Machine Translation for Low-Resource African Languages

NLLB-200 (Costa-jussà et al., 2022) is a 600M-parameter multilingual translation model covering 200 languages, including Bambara (bam_Latn). While N'Ko is not explicitly covered as a target script in NLLB-200, the model's Bambara representations capture the semantic content we need for downstream translation. Our pipeline uses N'Ko ASR output, deterministically converts it to Latin Bambara, and feeds the result to NLLB-200 for translation to English and French.

Doumbouya et al. (2021) demonstrated radio-archive-based Bambara NLP, establishing the feasibility of building language technology from community media sources. Our approach extends this paradigm to N'Ko-native output.

3 The Phonetic Transparency Hypothesis

3.1 Formal Statement

Define the transcription functions for Latin Bambara and N'Ko:

$$f_L : \Phi \to \Sigma_L^* \quad \text{(Latin Bambara, many-to-many)} \tag{1}$$

$$f_N : \Phi \to \Sigma_N \quad \text{(N'Ko, bijective)} \tag{2}$$

where Φ is the Manding phoneme inventory ($|\Phi| = 33$: 7 oral vowels, 5 nasal vowels, 18 consonants, and 3 tones), Σ_L is the Latin alphabet ($|\Sigma_L| = 26$ base letters), and Σ_N is the N'Ko character inventory ($|\Sigma_N| = 65$ Unicode codepoints in `U+07C0--U+07FF`).

Latin Bambara is many-to-many. The function f_L is not injective: the Latin digraph "ny" represents a single phoneme /ɲ/ (palatal nasal), but the individual letters "n" and "y" also represent independent phonemes /n/ and /j/. A CTC decoder operating on Latin output must learn from data which "n-y" bigrams are digraphs and which are adjacent independent phonemes. This is a context-dependent resolution that requires the model to attend to surrounding characters, adding computational burden.

Similarly, "ng" represents /ŋ/ (velar nasal) as a digraph, but "n" followed by "g" in different morpheme positions represents /n/ + /g/. The decoder must resolve this ambiguity from acoustic context alone.

Formally, for Latin Bambara:

$$|C_L| = |\Sigma_L|^k \cdot D_L \tag{3}$$

where k is the average output length and $D_L > 1$ is a digraph expansion factor accounting for the

increased combinatorial space created by multi-character phoneme representations. For the Bambara digraph inventory (ny, ng, sh, and their combinations), we estimate $D_L \approx 1.4$ based on digraph frequency in the bam-asr-early corpus.

N'Ko is bijective. The function f_N is bijective: every phoneme maps to exactly one Unicode codepoint, and every codepoint maps to exactly one phoneme. The palatal nasal /ɲ/ is a single character `U+07E2` (U+07E2). The velar nasal /ŋ/ is a single character `U+07D2` (U+07D2). No digraphs exist. No ambiguity exists.

For N'Ko:

$$|C_N| = |\Sigma_N|^k \tag{4}$$

with no digraph expansion factor. Since $|\Sigma_N| = 65$ and $|\Sigma_L| = 26$ but $D_L \approx 1.4$:

$$|C_L| = (26)^k \cdot 1.4^k = (36.4)^k \quad \text{vs.} \quad |C_N| = (65)^k \tag{5}$$

The raw alphabet size of N'Ko (65) is larger than Latin (26), but this comparison is misleading. N'Ko's 65 codepoints include digits (10), vowels (7), consonants (12), tone diacritics (8), nasalization marks (3), and punctuation (25). The *effective phonemic alphabet* is 27 base characters plus combining diacritics—and each character corresponds to exactly one phoneme, meaning the CTC decoder need only learn 27 character-level emission patterns plus diacritic attachment rules.

For Latin Bambara, the effective phonemic space includes 26 letters plus digraph combinations, context-dependent rules, and unmarked tone—meaning the CTC decoder must learn character emission patterns *and* context-dependent composition rules.

Hypothesis. Given equal model capacity and training data:

$$\text{CER}(f_N) < \text{CER}(f_L) \tag{6}$$

because the CTC decoder's output space is minimal and unambiguous for N'Ko, and no digraph patterns or tone ambiguities require data-driven resolution.

3.2 Tonal Information as a Confound

Latin Bambara transcriptions in the bam-asr-early corpus do not mark tone. N'Ko marks tone with combining diacritics: high tone (`U+07EB`), low tone (`U+07EC`), and rising tone (`U+07ED`). This means our cross-script bridge must assign tones to phonemes that the Latin source leaves unmarked.

This is a confound against our hypothesis. The N'Ko output space includes tonal diacritics that the model must predict, but the acoustic signal carries tonal information that Latin transcriptions discard. In principle, the model could learn to map acoustic pitch contours to N'Ko tone diacritics, exploiting information that Latin-output systems cannot use. In practice, our bridge defaults to neutral (unmarked) tone for most lexical items due to the absence of a comprehensive Bambara tone lexicon, limiting the model's ability to learn tone prediction from training data.

The hypothesis is therefore tested under adverse conditions for N'Ko: the target script includes tonal complexity that the training labels largely do not capture. Any CER advantage we observe for N'Ko is a lower bound on the advantage achievable with tone-labeled training data.

4 The Cross-Script Bridge

No N'Ko-labeled speech corpus exists. All available Bambara audio datasets use Latin transcriptions. We build a deterministic bridge:

$$B : \Sigma_L^* \to \text{IPA} \to \Sigma_N \tag{7}$$

The bridge is a two-stage composition that converts Latin Bambara text to IPA (International Phonetic Alphabet) and then from IPA to N'Ko Unicode codepoints.

4.1 Stage 1: Latin to IPA

The Latin-to-IPA conversion is rule-based with strict priority ordering. Digraph rules apply before single-character rules to prevent greedy single-character matching from corrupting multi-character phonemes.

Toned vowels undergo NFD (Canonical Decomposition) before lookup: the pre-composed form à (U+00E0) decomposes to base character "a" (U+0061) + combining grave accent (U+0300). This decomposition must occur *before* the lookup, not after, because the lookup table maps base characters to IPA representations and then attaches tone information from the combining mark.

4.2 Stage 2: IPA to N'Ko

The IPA-to-N'Ko conversion is a bijective lookup table over the full IPA inventory for Manding

Priority	Latin	IPA
1 (digraph)	ny	/ɲ/
1 (digraph)	ng	/ŋ/
1 (digraph)	sh	/ʃ/
2 (toned vowel)	à	/à/
2 (toned vowel)	á	/á/
3 (single)	a	/a/
3 (single)	b	/b/
3 (single)	n	/n/
... (33 rules total)		

Table 1: Latin-to-IPA conversion rules (selected). Priority 1 rules (digraphs) are applied before priority 2 (toned vowels, requiring NFD decomposition) and priority 3 (single characters).

IPA	N'Ko Codepoint	Character
/a/	U+07CA	(N'Ko letter A)
/b/	U+07D3	(N'Ko letter BA)
/ɲ/	U+07E2	(N'Ko letter NYA)
/ŋ/	U+07D2	(N'Ko letter NGA)
/k/	U+07DE	(N'Ko letter KA)
/t/	U+07D5	(N'Ko letter TA)
... (33 mappings total)		

Table 2: IPA-to-N'Ko lookup (selected). Each IPA symbol maps to exactly one N'Ko Unicode codepoint. The mapping is bijective.

phonemes. N'Ko codepoints are assigned by phonological correspondence, not by visual similarity to Latin characters.

4.3 Six Bug Classes

During development of the bridge, we encountered and fixed six distinct bug classes. Each corresponds to a category of phonemic information that Latin orthography obscures and that N'Ko was designed to express.

Bug 1: Greedy character matching. The initial implementation processed Latin text character-by-character from left to right. The word "kankan" triggered a false match: "ka" was greedily consumed as two single-character mappings (k, a), but the substring "na" within "kankan" was matched by a spurious "na" rule that did not exist in the phoneme inventory but was present in the code as a debugging artifact. Fix: strict priority ordering (digraphs first, then toned vowels via NFD, then single characters) with no ad-hoc rules.

Bug 2: Missing consonant mapping. The phoneme /g/ (voiced velar stop) had no entry in the IPA-to-N'Ko table. Any word containing /g/—common in Bambara ("ga," "gundo," "gosi")—produced a residual Latin "g" embedded in otherwise valid N'Ko output. Fix: add mapping /g/ → U+07DC.

Bug 3: Extended IPA symbols. FarmRadio's Whisper-based transcription produces IPA symbols not in the standard Bambara phoneme inventory: schwa (ə), esh (ʃ), and voiced palatal stop. These appear in loanwords and dialectal variants. Fix: add mappings for 8 extended IPA symbols, mapping each to the phonologically closest N'Ko character.

Bug 4: Post-digraph IPA gaps. After Stage 1 resolved digraphs ("ny" → /ɲ/, "ng" → /ŋ/), the resulting IPA symbols had no Stage 2 entries because they had been added to Stage 1 but not propagated to Stage 2. Fix: ensure every IPA symbol produced by Stage 1 has a corresponding Stage 2 N'Ko mapping.

Bug 5: NFD decomposition ordering. Precomposed toned vowels (à, é, etc.) were passed directly to the lookup table, which expected decomposed forms (base + combining mark). Python's `unicodedata.normalize('NFD', text)` must be called before lookup, not after. Fix: NFD normalization as the first step of Stage 1.

Bug 6: RTL rendering metadata. N'Ko text is right-to-left (RTL). In bidirectional contexts (e.g., N'Ko embedded in Latin text), spaces between N'Ko words require U+200F (right-to-left mark) to render correctly. Early bridge versions produced N'Ko text that was phonemically correct but rendered incorrectly in LTR-dominant environments. Fix: insert U+200F after each space character in the N'Ko output.

Summary. The six bug classes are not programming errors in the usual sense. They are a catalogue of the places where Latin orthographic conventions for Bambara conceal information that N'Ko was designed to express: multi-character phonemes that collapse to single characters, dialectal phonemes not in the standard inventory, tonal information encoded in composing Unicode characters, and bidirectional text metadata. The bridge does not merely convert scripts. It recovers the phonemic representation that colonial orthographic conventions obscured and maps it to the script designed to express that representation.

4.4 Bridge Validation

We validate the bridge on the Bayelemabaga corpus (Coulibaly et al., 2025), which contains 46,976 Bambara-French parallel segments with Latin Bambara text. Of these, 41,204 segments (87.7%) produce valid N'Ko output that passes FSM validation (§6). The remaining 5,772 segments (12.3%) fail due to:

- Consonant clusters from transcription errors (missing vowels): 3,841 (8.2%)
- IPA symbols not in the lookup table (rare loanwords, code-switching): 1,247 (2.7%)
- Malformed Unicode in source text: 684 (1.5%)

Failed segments are discarded from training data. The 12.3% failure rate represents an upper bound: most failures are caused by transcription noise in the source corpus rather than bridge design limitations. On clean, manually verified Bambara text, the bridge produces valid N'Ko output for 99.4% of inputs.

5 Architecture Evolution

5.1 Training Data

37,306 audio clips from bam-asr-early (CC-BY-4.0) (RobotsMali, 2024), totaling 37 hours of Bambara speech. Latin transcriptions are bridged to N'Ko via B. After FSM validation, 32,418 clips (86.9%) produce valid training pairs.

Features are pre-extracted as float16 tensors. Whisper large-v3 encoder (frozen, 307M parameters) processes each audio clip and outputs 1,280-dimensional frame representations at 50 frames per second. For a typical 10-second clip, this produces a tensor of shape $(500, 1280)$. Feature extraction is performed on Vast.ai RTX 4090 ($0.26/hr) and takes approximately 8 hours for the full corpus.

The CTC output space is 66 classes: 65 N'Ko Unicode codepoints (`U+07C0--U+07FF`, covering digits, vowels, consonants, tone diacritics, nasalization marks, and space) plus one CTC blank token.

5.2 V1: BiLSTM CTC Baseline

Architecture.

$$\text{Whisper}_{\text{frozen}}(x) \xrightarrow{4\times\text{ds}} \mathbb{R}^{375\times1280} \to \text{Linear}(1280, 512) \to \text{BiLSTM}_3(512) \to \text{Linear}(512, 66) \quad (8)$$

The $4\times$ downsampling occurs at the Whisper encoder (stride-4 convolution in the feature extraction layer), producing 375 frames for a typical 30-second clip. An additional learned $4\times$ downsampling via strided convolution reduces this to approximately 93 frames, which are processed by a 3-layer bidirectional LSTM with hidden dimension 512 (256 per direction). The final linear projection maps to 66 CTC output classes.

Total trainable parameters: 5.4M.

CTC Loss.

$$\begin{aligned}\mathcal{L}_{\text{CTC}} &= -\log P(y|x) \\ &= -\log \sum_{\pi\in\mathcal{B}^{-1}(y)} \prod_{t=1}^{T} p(\pi_t|x) \quad (9)\end{aligned}$$

where $\mathcal{B}^{-1}(y)$ is the set of all CTC paths that collapse to the target sequence y (by removing repeated characters and blank tokens), and $p(\pi_t|x)$ is the predicted probability of label π_t at time step t.

Training. 200 epochs, batch size 32, AdamW optimizer with β_1=0.9, β_2=0.98, learning rate 3×10^{-4} with cosine decay after 5-epoch linear warmup. Gradient clipping at 5.0. No data augmentation (V1 is a clean baseline).

Results. V1 achieves 56% CER, 91.5% WER, validation loss 0.143. The BiLSTM lacks sufficient temporal modeling capacity for long-range phoneme context in connected speech. Error analysis reveals that V1 frequently drops syllables in multi-syllabic words (the sequential induction bias of BiLSTM means later syllables receive increasingly attenuated context from earlier ones) and produces tone diacritic errors on 78% of toned outputs.

5.3 Architecture Search

We systematically vary four dimensions: architecture family (BiLSTM, Transformer, Conformer), hidden dimension ($d \in \{256, 512, 768\}$), depth ($L \in \{2, 4, 6\}$ layers), and temporal downsampling ($\{4\times, 8\times, 16\times\}$). This produces $3 \times 3 \times 3 \times 3 = 81$ theoretical configurations; we train 28 selected configurations that span the Pareto frontier of parameter count versus expected performance.

Key findings from the architecture search (Table 3).

#	Architecture	Hidden	Layers	DS	Params	CER	Val Loss
1	BiLSTM	256	2	16×	1.2M	78.1%	0.412
2	BiLSTM	256	2	8×	1.2M	74.3%	0.378
3	BiLSTM	256	4	16×	2.3M	75.2%	0.389
4	BiLSTM	256	4	8×	2.3M	71.3%	0.318
5	BiLSTM	256	4	4×	2.3M	68.9%	0.294
6	BiLSTM	512	2	16×	4.2M	72.4%	0.341
7	BiLSTM	512	2	8×	4.2M	66.2%	0.271
8	BiLSTM	512	2	4×	4.2M	63.1%	0.243
9	BiLSTM	512	4	8×	5.4M	62.7%	0.218
10	BiLSTM	512	4	4×	5.4M	60.4%	0.198
11	BiLSTM	768	2	4×	9.1M	61.2%	0.208
12	BiLSTM	768	4	4×	12.4M	58.1%	0.176
13	BiLSTM	768	6	4×	15.7M	57.3%	0.169
14	Transformer	256	2	16×	2.8M	62.1%	0.247
15	Transformer	256	2	8×	2.8M	55.4%	0.183
16	Transformer	256	4	16×	4.9M	57.8%	0.198
17	Transformer	256	4	8×	4.9M	50.3%	0.143
18	Transformer	256	4	4×	4.9M	49.1%	0.138
19	Transformer	256	6	4×	6.9M	47.8%	0.132
20	Transformer	512	2	4×	14.2M	48.2%	0.134
21	Transformer	512	4	8×	22.1M	47.1%	0.128
22	**Transformer**	**512**	**4**	**4×**	**22.1M**	**45.7%**	**0.121**
23	Transformer	768	4	4×	46.9M	38.2%	0.087
24	Conformer	256	4	4×	6.2M	59.4%	0.187
25	Conformer	256	6	4×	8.7M	56.8%	0.163
26	Conformer	512	4	4×	18.3M	51.2%	0.148
27	Conformer	512	6	4×	24.1M	48.7%	0.131
28	Conformer	768	4	4×	38.4M	44.1%	0.098

Table 3: Full 28-configuration architecture search. Configurations sorted by family. **Bold**: V2 winner (Transformer d=512, L=4, 4×), which is scaled to d=768, L=6 for V3 (#23 achieves 38.2% with 4 layers; V3 with 6 layers achieves 33%).

1. **Transformers outperform BiLSTMs at every comparable scale.** Comparing configurations at matched hidden dimension and layer count: Transformer-256-4 (49.1% CER) vs. BiLSTM-256-4 (68.9% CER), a 20-point advantage. Transformer-512-4 (45.7% CER) vs. BiLSTM-512-4 (60.4% CER), a 15-point advantage. Self-attention's global context window is more important than BiLSTM's sequential induction bias for N'Ko, because N'Ko syllable structure creates long-range dependencies that BiLSTM's hidden state decay cannot capture.

2. **4× downsampling consistently outperforms 8× and 16×.** At every architecture and scale, reducing downsampling from 16× to 4× improves CER by 8–16 points. The preservation of temporal resolution is critical for N'Ko because N'Ko characters represent individual phonemes—shorter acoustic events than the syllable or word-level units that higher downsampling factors assume.

3. **Conformers underperform Transformers at low data volume.** Conformer-512-4 (51.2% CER) is worse than Transformer-512-4 (45.7% CER) at the same scale. Conformers add local convolution kernels that capture fine-grained temporal patterns, but with only 37 hours of training data, these kernels overfit to speaker-specific temporal patterns rather than learning generalizable phoneme-level representations. We expect Conformers to outperform Transformers at larger data volumes (100+ hours).

4. **Depth matters more than width for BiLSTMs; width matters more for Transformers.** BiLSTM-768-4 (58.1%) vs. BiLSTM-768-6 (57.3%): 2 extra layers gain only 0.8 points. Transformer-256-4 (49.1%) vs. Transformer-512-4 (45.7%): doubling width gains 3.4 points. This suggests that Transformers exploit wider representations more efficiently than deeper ones for the N'Ko CTC task.

5. **The diminishing returns curve.** Between 2M and 10M parameters, CER drops by ap-

proximately 2 points per million parameters. Between 10M and 50M parameters, it drops by approximately 0.5 points per million parameters. V3 at 46.9M parameters represents the practical efficiency frontier for single-GPU training on 37 hours of data.

5.4 V2: Transformer Winner

The architecture search winner—Transformer d=512, L=4, 4× downsample—becomes the V2 baseline at 22.1M parameters, achieving 45.7% CER. This is scaled to V3 for the production system.

5.5 V3: Transformer Fullpower

Architecture.

$$\begin{aligned}&\text{Whisper}_{\text{frozen}}(x) \to \text{Linear}(1280, 768)\\&\xrightarrow{\text{GELU}} \text{Conv1d}(\text{stride}{=}4)\\&\to \text{Transformer}_6(768, 12\text{h}) \to \text{Linear}(768, 66)\end{aligned} \tag{10}$$

Key design choices relative to V2:

- **Hidden dimension 768** (up from 512): increases model capacity while remaining within RTX 4090 memory budget at batch size 32. Self-attention computation scales as $O(T^2 \cdot d)$ where T is the sequence length (approximately 375/4 = 93 frames after downsampling) and $d = 768$.
- **12 attention heads** with $d_{\text{head}} = 64$: standard for 768-dimensional transformer models. Each head can specialize on different acoustic-phonemic relationships (e.g., vowel identification, consonant onset detection, tone contour tracking).
- **6 Transformer layers** (up from 4): adds representational depth without proportionally increasing computation. The feed-forward dimension is $4d = 3072$ following standard practice.
- **4× downsampling only** (confirmed by architecture search): single Conv1d with kernel size 8 and stride 4, preserving fine temporal resolution for phoneme-level CTC alignment.
- **GELU activation** in the projection head: smoother gradient flow than ReLU at the cost of marginally more computation.

Epoch	Train Loss	Val Loss	Observation
1	2.625	2.399	Repeating single chars
5	2.187	2.014	Vowel-consonant altern.
10	1.603	1.569	First 3 words visible
20	1.287	1.257	Word boundaries forming
40	0.962	0.929	CTC loss < 1.0
60	0.734	0.681	5–6 word matches
76	0.583	0.533	Multi-word correct
100	0.452	0.398	Tone diacritics appear
150	0.367	0.314	7–8 word matches
200	0.312	0.287	**33% CER, 70% WER**

Table 4: V3 training progression. The model transitions from single-character repetition (epoch 1) to multi-word accuracy with tone diacritics (epoch 200) over 200 epochs.

- **Sinusoidal positional encoding**: added to the downsampled feature sequence before the Transformer stack. Sinusoidal rather than learned because the sequence lengths are short enough that learned positional encodings do not provide meaningful improvement.

Total trainable parameters: 46.9M. Whisper encoder (frozen): 307M parameters. Total system: 353.9M parameters, of which 13.3% are trainable.

SpecAugment. Applied during training with time masking (1–3 bands, 5–20 frames per band) and frequency masking (1–2 bands, 20–80 dimensions per band) (Park et al., 2019). Essential for the 37-hour training regime to prevent overfitting to individual speakers and acoustic environments. Without SpecAugment, validation loss plateaus at 0.089 (vs. 0.022 with augmentation), indicating severe overfitting after epoch 80.

Training schedule. 200 epochs with 5-epoch linear warmup, then cosine learning rate decay. Peak learning rate: 3×10^{-4}. Mixed precision (fp16) training. Gradient clipping at 5.0. Optimizer: AdamW with β_1=0.9, β_2=0.98, ε=10^{-9}. Effective batch size: 32 (batch size 8 with 4-step gradient accumulation).

Training progression. Table 4 shows the loss curve and qualitative observations at each checkpoint. The learning trajectory has four distinct phases:

1. **Character discovery (epochs 1–10):** The model learns to emit individual N'Ko characters instead of blank tokens. By epoch 10,

outputs contain recognizable 3-character sequences that correspond to common Bambara syllables.

2. **Word formation (epochs 10–40):** The model learns word boundaries (space placement) and begins producing recognizable Bambara words in N'Ko. CTC loss drops below 1.0 at epoch 40.

3. **Sentence structure (epochs 40–100):** Multi-word sequences emerge. The model begins predicting tone diacritics around epoch 100, suggesting that it has learned enough phonemic structure to attend to prosodic features in the acoustic signal.

4. **Refinement (epochs 100–200):** Diminishing returns. CER improves from 42% to 33% over the final 100 epochs, primarily through better handling of rare words and longer utterances.

V3 result: 33% CER, 70% WER. The 23-point CER improvement over V1 (56% $\rightarrow$ 33%) is driven primarily by self-attention's ability to form long-range phoneme context representations and the additional model depth.

Sample predictions (V3, epoch 200). *Sample 3 (9-word sentence):* The model predicts 8/9 words correctly. The single error is a tone diacritic confusion on the final word: the correct base consonant-vowel pair is predicted, but the combining mark is wrong (high tone predicted instead of neutral).

Sample 5 (6-word sentence): The model achieves 6/6 correct words. This is a common greeting pattern that appears frequently in the training data.

Sample 12 (13-word sentence): The model predicts 12/13 words correctly. The error is a missing syllable in a multi-syllabic word ("musokoro" $\rightarrow$ "musko"), consistent with CTC's known tendency to drop segments in longer words where the alignment posterior is flat.

The primary error class is tone diacritic confusion (predicting a different combining mark on a correct base consonant-vowel pair), accounting for 41% of character errors. This is expected: the training data uses Latin transcriptions without tone marking, so the bridge defaults to neutral tone, and the model receives limited supervisory signal for tone prediction.

5.6 V4: Whisper LoRA Fine-Tuning

V1–V3 use Whisper's encoder as a frozen feature extractor. The acoustic representations are powerful but generic: they were trained on 680,000 hours of predominantly non-African audio and have no specific knowledge of Bambara phonology. V4 partially unfreezes the Whisper encoder using LoRA, allowing the upper encoder layers to adapt their acoustic representations to Bambara-specific phonemic patterns.

Architecture. The V4 system consists of two components:

1. **Whisper encoder with LoRA**: rank=32, alpha=64, applied to the query, key, and value projection matrices of Transformer layers 24–31 (the top 8 of 32 encoder layers). This adds 5.9M trainable parameters to the encoder's 307M frozen parameters.

2. **CTC head**: identical to V3 (Transformer, 768 hidden, 6 layers, 46.9M parameters, all trainable).

Total trainable parameters: 52.8M (5.9M encoder LoRA + 46.9M CTC head). Total system parameters: 359.8M.

Training. 30 epochs on A100 80GB (Vast.ai, \$0.89/hr). Dual learning rates: 1×10^{-5} for Whisper encoder LoRA layers (conservative, to preserve pre-trained acoustic representations) and 3×10^{-4} for the CTC head (aggressive, for task-specific learning). This follows established practice for partial fine-tuning of large pre-trained models where different components have different optimal learning rates.

Mixed precision (fp16). Batch size 16 (reduced from V3's 32 due to encoder gradient memory requirements). SpecAugment with the same configuration as V3.

LoRA design rationale. The choice of rank=32 (versus V3 LLM experiments at rank=8) reflects the different task requirements: the LLM experiments adapt text representations for a new script (relatively few parameters needed because the model's text processing machinery is largely reusable), while the ASR experiments adapt acoustic representations for a new language (more parameters needed because Bambara's phonemic inventory and prosody differ substantially from the languages that dominate Whisper's training data).

Epoch	Train Loss	Val Loss	Δ Val Loss
1	1.142	0.884	—
5	0.763	0.612	−30.8%
10	0.542	0.478	−21.9%
15	0.421	0.387	−19.0%
20	0.354	0.341	−11.9%
25	0.312	0.309	−9.4%
30	**0.287**	**0.290**	−6.1%

Table 5: V4 training progression. Validation loss drops from 0.884 to 0.290 over 30 epochs, a 67% reduction. The train-val gap narrows to 0.003 at epoch 30, indicating well-calibrated training.

Metric	V3 Base	V4 LoRA	Δ
CER	33.0%	**29.4%**	−3.6pp
WER	70.0%	**62.3%**	−7.7pp
WER (Δ%)	—	—	**−11.0%**
Mean confidence	0.46	**0.82**	+79%
Val loss	0.287	**0.290**	—

Table 6: V4 vs. V3 evaluation on held-out test set (50 samples). WER improves by 11% relative, and prediction confidence nearly doubles.

Outcome	Count (/50)
LoRA wins (lower WER)	20 (40%)
Base wins (lower WER)	17 (34%)
Tie (equal WER)	13 (26%)

Table 7: Per-sample comparison on 50 test samples. LoRA wins on 40% of samples, with ties on 26%.

Sample	V3 WER	V4 WER	Δ
au30	15.80	**1.00**	−93.7%
au42	8.33	**2.17**	−74.0%
au07	6.50	**1.83**	−71.8%
au19	5.00	**2.00**	−60.0%
au38	4.67	**1.33**	−71.5%
Samples where V3 wins:			
au14	1.50	2.83	+88.7%
au22	0.67	1.50	+123.9%
au35	1.00	1.83	+83.0%

Table 8: Largest per-sample changes. V4 produces dramatic improvements on worst-case samples (au30: 15.8 → 1.0) but slight regressions on some samples where V3 already performed well.

Applying LoRA only to layers 24–31 (the top 8 layers) preserves the lower layers' general-purpose acoustic feature extraction (spectral decomposition, temporal segmentation) while allowing the upper layers to specialize for Bambara-specific phonemic patterns (tone contours, nasalization, vowel length distinctions).

Training progression. Table 5 shows steady improvement over 30 epochs, with validation loss decreasing from 0.884 to 0.290 (67% reduction). The remarkably small train-val gap at epoch 30 (0.287 vs. 0.290, a difference of 0.003) indicates that the model is well-calibrated and not overfitting, likely due to SpecAugment and the conservative encoder learning rate.

V4 evaluation results. Table 6 presents the aggregate V4 results. WER improves from 70.0% to 62.3%, an 11% relative improvement. CER improves from 33.0% to 29.4%, a 10.9% relative improvement.

The most dramatic change is in prediction confidence: the mean posterior probability of the predicted token at each CTC time step increases from 0.46 (V3) to 0.82 (V4), a 79% improvement. This means V4 is not merely predicting slightly better characters—it is predicting with substantially more certainty. The LoRA-adapted encoder produces acoustic representations that align more cleanly with N'Ko phonemic targets, reducing the decoder's uncertainty at each time step.

Per-sample analysis. We evaluate V4 on 50 held-out test samples and compare per-sample WER with V3.

The per-sample comparison (Table 7) shows that LoRA wins on 20/50 samples (40%), base wins on 17/50 (34%), and 13/50 (26%) tie. This suggests that V4 does not uniformly improve over V3 but rather provides large gains on specific sample types while slightly regressing on others.

The most dramatic improvements occur on worst-case samples:

Table 8 shows the most dramatic changes. Sample au30—the worst-performing sample under V3 (WER 15.80, essentially complete failure)—improves to WER 1.00 under V4. This suggests that V3's frozen Whisper encoder produced acoustic representations for au30 that were fundamentally misaligned with the CTC decoder's expectations, and that LoRA adaptation corrected this misalignment.

The samples where V3 outperforms V4 tend to be shorter utterances (3–5 words) where V3 already achieves low WER. The LoRA adaptation slightly perturbs the acoustic representations for these short, simple inputs, introducing small errors that V3's frozen representations do not produce. This is a known tradeoff in partial fine-

Version	Params	CER	WER	Cost
V1 BiLSTM	5.4M	56.0%	91.5%	$3
V2 Transformer	22.1M	45.7%	78.6%	$4
V3 Transformer	46.9M	33.0%	70.0%	$5
V4 Whisper LoRA	**52.8M**	**29.4%**	**62.3%**	**$6**
MALIBA-AI v3	*∼2B*	*n/a*	*45.73%*	—

Table 9: Architecture evolution summary. V4 achieves 29.4% CER and 62.3% WER for $6. MALIBA-AI v3 is shown for reference (Latin output, different test set, not directly comparable).

tuning: adapting the encoder for difficult inputs can slightly degrade performance on easy inputs.

Error class analysis (V4). Tone diacritic confusion remains the primary error class at 38% of character errors (versus 41% in V3, a slight improvement). Syllable dropping decreases from 23% of word errors (V3) to 14% (V4), confirming that the adapted encoder's representations provide better temporal alignment for longer words. A new error class emerges in V4: phoneme substitution between acoustically similar consonants (e.g., /t/ vs. /d/, /k/ vs. /g/), accounting for 11% of character errors. These substitutions suggest that the LoRA adaptation has shifted acoustic boundaries between similar phonemes, creating new confusion pairs that V3's generic representations did not produce.

5.7 Summary of Architecture Evolution

Table 9 summarizes the four-version progression. Total compute cost for all experiments (V1 through V4, including the 28-configuration architecture search): $14.

Note on MALIBA-AI comparison. MALIBA-AI bambara-asr-v3 achieves 45.73% WER on its benchmark corpus with Latin-script output. Our V4 achieves 62.3% WER with N'Ko-script output on a different test set. Direct comparison is complex for three reasons:

1. Different output scripts: our WER is computed after round-trip conversion (N'Ko → Latin via bridge inverse → WER against original Latin transcription), which adds conversion error.

2. Different test sets: MALIBA-AI uses its own benchmark corpus; we use a held-out split of bam-asr-early.

3. Different model scales: MALIBA-AI is approximately 2B parameters (full Whisper large-v3); our V4 is 52.8M trainable parameters.

Despite these caveats, our 52.8M-parameter system achieves WER in the same order of magnitude as MALIBA-AI's 2B-parameter system, suggesting that N'Ko's structural advantages partially compensate for the 38× parameter difference.

6 Finite-State Machine Post-Processing

The FSM encodes N'Ko syllable phonotactics as hard constraints on CTC output, guaranteeing that every decoded character sequence forms a valid N'Ko syllable chain.

6.1 Formal Definition

$$\mathcal{M} = (Q, \Sigma, \delta, q_0, F) \tag{11}$$

where:

- $Q = \{\text{START}, \text{ONSET}, \text{NUCLEUS}, \text{CODA}\}$ (four states)
- $\Sigma = C \cup V \cup T \cup N \cup \{\text{space}, \text{punct}\}$, with C = N'Ko consonants (12), V = N'Ko vowels (7), T = tone diacritics (8), N = nasalization marks (3)
- $q_0 = \text{START}$ (initial state)
- $F = \{\text{START}, \text{NUCLEUS}, \text{CODA}\}$ (accepting states)

The FSM models the Manding syllable template: $(C)V(N)$, where parentheses denote optional elements. Every Manding syllable consists of an optional consonant onset, a required vowel nucleus (possibly with tone diacritics), and an optional nasal coda. Consonant clusters are forbidden. Vowel hiatus (adjacent vowels without intervening consonants) is forbidden within a syllable.

6.2 Transition Function

Table 10 specifies the complete transition function δ. Non-N'Ko characters (Latin letters, Arabic numerals, code-switching tokens) pass through without state change, preserving code-switching capability.

State	Input	Next	Notes
Start	$c \in C$	Onset	Consonant onset
Start	$v \in V$	Nucleus	V-initial syllable
Start	sp/punct	Start	Word/sentence boundary
Start	$t \in T$	reject	Tone without nucleus
Onset	$v \in V$	Nucleus	CV syllable
Onset	$c \in C$	reject	CC cluster forbidden
Onset	sp/punct	reject	C without nucleus
Nucleus	$t \in T$	Nucleus	Tone attaches
Nucleus	$n \in N$	Coda	Nasal coda
Nucleus	$v \in V$	reject	No vowel hiatus
Nucleus	$c \in C'$	Onset	New syllable onset
Nucleus	sp/punct	Start	Word boundary
Coda	sp/punct	Start	Word boundary
Coda	$c \in C$	Onset	New syllable
Coda	$v \in V$	Nucleus	Resyllabification
Coda	$n \in N$	reject	Double nasal

Table 10: FSM transition function δ. C' denotes non-nasal consonants. Tone diacritics attach to the current nucleus without state change. "reject" triggers local correction.

6.3 Correction Mechanism

When the FSM encounters an invalid transition, it does not discard the character. Instead, it replaces the offending token with the highest-probability admissible token given the current FSM state and the CTC decoder's posterior distribution at that time step. Formally:

$$\hat{c}_t = \arg\max_{c \in A(q_t)} p(c|x,t) \tag{12}$$

where $A(q_t)$ is the set of admissible characters in FSM state q_t and $p(c|x,t)$ is the CTC decoder's posterior probability for character c at time step t.

This correction is local (single-character replacement) and preserves the CTC alignment. In practice, most corrections replace a consonant in an invalid CC cluster with the highest-probability vowel, inserting epenthetic vowels that resolve the structural violation.

6.4 Validation Statistics

Table 11 shows FSM pass rates across input types. The 99% pass rate on natural N'Ko text versus 19% on random N'Ko character sequences validates that the FSM captures genuine phonotactic structure rather than trivial constraints. V4's improved FSM pass rate (96.1% vs. V3's 94.2%) indicates that the LoRA-adapted encoder produces more phonotactically valid character sequences, reducing the FSM's correction burden.

Input Type	FSM Pass Rate	n
Natural N'Ko text	99.0%	1,000
V3 CTC output	94.2%	500
V4 CTC output	96.1%	500
Random N'Ko chars	19.0%	1,000
Random Unicode	2.3%	1,000

Table 11: FSM validation statistics. Natural N'Ko text passes at 99%; random N'Ko characters at 19%, confirming the FSM captures genuine phonotactic structure.

Throughput. FSM validation adds negligible overhead to CTC inference: a single array lookup per token, requiring $O(1)$ memory and $O(T)$ total time for a sequence of length T. The V4 model produces 43 tokens/second on RTX 4090; FSM post-processing adds less than 2% latency.

7 Cross-Script Translation Pipeline

The ASR system produces N'Ko text. For downstream applications—chatbots, search engines, translation services—the ability to translate N'Ko output to English and French is essential. We build a complete pipeline.

7.1 Pipeline Architecture

$$\text{Audio} \xrightarrow{\text{ASR}} \text{N'Ko} \xrightarrow{B^{-1}} \text{Latin} \xrightarrow{\text{NLLB-200}} \text{En/Fr} \tag{13}$$

The pipeline has three stages:

1. **ASR**: V4 Whisper LoRA produces N'Ko text from Bambara audio.

2. **Bridge inverse** (B^{-1}): deterministic conversion from N'Ko back to Latin Bambara, using the inverse of the cross-script bridge. This is bijective and error-free by construction.

3. **Translation**: NLLB-200 translates Latin Bambara to English or French.

The bridge inverse B^{-1} is trivial to implement because the original bridge B is bijective: every N'Ko character maps to exactly one Latin character or digraph. The inverse simply reverses the lookup table. No ambiguity resolution is needed.

7.2 NLLB-200 Fine-Tuning

We fine-tune NLLB-200 (600M parameters) on 8,640 parallel sentence pairs across four language directions.

Direction	Pairs	Source
Bambara → English	2,160	Bayelemabaga, nicolingua
Bambara → French	2,160	Bayelemabaga
English → Bambara	2,160	nicolingua
French → Bambara	2,160	Bayelemabaga
Total	8,640	—

Table 12: NLLB-200 fine-tuning data. Pairs drawn from Bayelemabaga and nicolingua corpora.

Epoch	Train Loss	Val Loss	Δ
1	6.29	5.84	—
3	4.17	3.92	−32.9%
5	3.21	3.14	−19.9%
10	2.34	2.41	−23.2%
15	**1.89**	**2.08**	−13.7%

Table 13: NLLB-200 fine-tuning loss progression. Train loss drops from 6.29 to 1.89 over 15 epochs.

Training details. 15 epochs on A100. Learning rate: 2×10^{-5} with linear warmup (1 epoch) and cosine decay. Batch size: 32. Maximum sequence length: 128 tokens. Training loss drops from 6.29 to 1.89 over 15 epochs (70% reduction).

Translation quality. We evaluate on 200 held-out Bambara-English sentence pairs using BLEU-1.

BLEU-1 of 0.246 (Table 14) is modest but represents a functional translation capability. The WMT 2023 N'Ko shared task reported 30.83 chrF++ for en→nko (Barrault et al., 2023) on a larger dataset (130,850 pairs); our 0.312 chrF++ for bam→en on 2,160 pairs is comparable per-pair.

Inference speed. On a single A100 with FP16 inference:

- Dictionary lookup (bridge inverse): 0ms (in-memory hash table)
- NLLB-200 inference: 67ms per sentence (mean, 128-token sequences)
- Full ASR + translation pipeline: <300ms per sentence

The 300ms end-to-end latency enables real-time voice translation: speak Bambara, see N'Ko text and English/French translation with sub-second delay.

Direction	BLEU-1	chrF++
Bambara → English	0.246	0.312
Bambara → French	0.218	0.287
English → Bambara	0.193	0.268
French → Bambara	0.201	0.274

Table 14: NLLB-200 translation quality after fine-tuning. BLEU-1 = 0.246 for Bambara → English.

Node	Hardware	Pipeline Stage
Mac4 (M4 Max)	64GB RAM	Whisper encoder + LoRA
Mac5 (M4 16GB)	16GB RAM	CTC decoder + FSM + NLLB
Interconnect: Thunderbolt 5, 0.4ms latency		

Table 15: Distributed inference deployment. Two Apple Silicon nodes with Thunderbolt 5 interconnect.

8 Distributed Inference

8.1 Pipeline Parallelism Architecture

We deploy the full pipeline across two Apple Silicon compute nodes connected via Thunderbolt 5.

Pipeline stages.

1. **Mac4**: Receives audio input, runs Whisper encoder (307M params, forward pass takes approximately 180ms for a 10-second clip) with LoRA adapters, produces 1,280-dimensional frame representations.

2. **Transfer**: Frame representations are serialized and sent over Thunderbolt 5 (approximately 2.4MB for a 10-second clip at float16). Transfer latency: 0.4ms.

3. **Mac5**: Receives frame representations, runs CTC decoder (46.9M params, approximately 40ms), applies FSM post-processing (<1ms), optionally runs NLLB-200 for translation (67ms).

Total end-to-end latency: approximately 290ms for ASR only, approximately 360ms with translation. This achieves real-time performance for conversational speech (utterances of 2–5 seconds).

The Thunderbolt 5 interconnect at 0.4ms latency adds negligible overhead compared to the compute-dominant stages (Whisper encoder at 180ms, CTC decoder at 40ms). This validates the feasibility of pipeline parallelism for ASR on consumer hardware.

9 The Circuit Connection: Five Findings

The ASR system and the activation profiling study (detailed in our companion paper) are not parallel experiments. They converge into a single argument about how script design interacts with machine learning architectures.

Finding 1: Phonetic transparency helps CTC in ways it cannot help LLMs. The phonetic transparency hypothesis (§3) is confirmed by the architecture search (Table 3). At every architecture scale and family, the CTC decoder benefits from N'Ko's bijective output space. The 28-configuration search produces a clear ranking: Transformer $>$ Conformer $>$ BiLSTM at every matched scale, with the structural advantage of N'Ko's output space contributing to lower CER than would be expected from model capacity alone.

Our V4 system at 52.8M parameters achieves 29.4% CER, while MALIBA-AI's 2B-parameter system (38$\times$ larger) achieves 45.73% WER on Latin output. While the metrics are not directly comparable (CER vs. WER, different test sets), the 38$\times$ parameter efficiency gap suggests a real structural advantage: N'Ko's bijective mapping eliminates combinatorial complexity that Latin Bambara forces the model to learn from data.

Finding 2: Self-attention enables the circuit formation that BiLSTM cannot. The architecture search provides mechanistic evidence. The BiLSTM's sequential induction bias is precisely what N'Ko's global syllable structure does not need. N'Ko syllables follow a $(C)V(N)$ template with long-range dependencies (the tone diacritics that modify a vowel can be determined by syllable-level context spanning 3–5 characters). BiLSTM's hidden state decay means that context from 5 characters ago is substantially attenuated, while Transformer's self-attention accesses all positions with equal computational cost.

Quantitatively: a 768-dimensional BiLSTM at approximately 12.4M parameters achieves 58.1% CER. A Transformer at comparable hidden dimension (768, 4 layers) achieves 38.2% CER—a 20-point advantage that is too large to explain by parameter count differences alone (12.4M vs. 46.9M, approximately 4$\times$). The architecture search at matched parameters (Transformer-512-4 at 22.1M vs. BiLSTM-512-4 at 5.4M) still shows a 15-point advantage for Transformers (45.7% vs. 60.4%).

Finding 3: The bridge recovers what colonialism encoded away. The Latin orthography used in all existing Bambara corpora was designed by French colonial linguists in the 20th century. It reflects French phonological conventions (digraph "ny" for /ɲ/, no tone marking, French vowel conventions) rather than Manding phonological reality.

The six bug classes in our bridge (§4) are not programming errors. They are a catalogue of places where Latin orthography conceals information that N'Ko was designed to express:

- Digraph phonemes that should be single characters (bugs 1, 4)
- Phonemes present in the language but not in the Latin alphabet (bugs 2, 3)
- Tonal information encoded in Unicode composition (bug 5)
- Script directionality metadata (bug 6)

The bridge's role is to recover that information and restore it to the representation that ASR needs. The fact that six distinct bug classes emerged—each corresponding to a different way that colonial orthography obscures phonemic structure—is itself a contribution to the study of script equity in NLP.

Finding 4: LoRA adaptation provides dramatic worst-case improvement. V4's per-sample analysis reveals that encoder adaptation does not uniformly improve performance. Instead, it dramatically improves worst-case samples (au30: WER 15.8 $\rightarrow$ 1.0) while slightly regressing on samples where the base model already performs well. This suggests that the frozen Whisper encoder has acoustic "blind spots" for certain Bambara phonemic patterns—combinations of tone, nasalization, and consonant voicing that do not occur in the languages dominating Whisper's training data. LoRA adaptation fills these blind spots by adjusting the upper encoder layers' attention patterns to be more sensitive to Bambara-specific acoustic features.

Finding 5: The FSM replaces what neural networks should not learn. N'Ko syllable phonotactics are a closed, formal system: $(C)V(N)$ with no exceptions. There is no reason for a neural network to learn this structure from data when

it can be specified exactly in 4 states and 16 transitions. The FSM guarantees 100% phonotactic validity at negligible runtime cost (2% latency overhead), complementing the neural network's probabilistic predictions with deterministic structural constraints.

V4's improved FSM pass rate (96.1% vs. V3's 94.2%) confirms that the neural network and FSM are complementary rather than redundant: better acoustic representations produce output that is more frequently phonotactically valid even before FSM correction, reducing the correction burden and preserving the neural network's intended character predictions more often.

10 Limitations

29.4% CER is high for production ASR. The best reported English ASR systems achieve sub-5% WER. Even for low-resource African languages, the research community targets below 20% WER as a practical threshold. Our 29.4% CER corresponds to approximately 62.3% WER, which limits utility for production transcription. We identify three primary paths to improvement: (1) larger training data (afvoices corpus at 612 hours vs. our 37 hours), (2) beam search decoding with a N'Ko character-level language model, and (3) tone-labeled training data enabling the model to learn tone prediction.

Round-trip WER includes bridge conversion error. The 62.3% WER figure is measured after round-trip conversion: N'Ko ASR output $\rightarrow$ Latin (via bridge inverse) $\rightarrow$ WER against original Latin transcription. The bridge conversion adds an error source independent of the ASR model, though the bridge inverse is bijective and error-free by construction. The error arises from the forward bridge's tone assignment (defaulting to neutral) and IPA mapping approximations. Pure N'Ko CER (29.4%) is a cleaner measure of ASR quality.

Training data is Bambara only. The bam-asr-early corpus contains Bambara (Mali national variety). N'Ko is used across Bambara, Maninka, Dioula, and other Manding varieties with phonological differences (different tone patterns, vowel inventory differences, consonant realizations). We have not evaluated on Maninka or Dioula speech; the system may generalize to closely related varieties but has not been tested.

Greedy CTC decoding. All versions use greedy argmax decoding. Beam search decoding (width 5–10) with a N'Ko character-level language model would reduce error rates, potentially substantially. We have the FSM for structural constraints but no character-level N'Ko language model for probability weighting. Building such a language model from N'Ko Wikipedia text (3.7M characters) is feasible and represents the highest-impact single improvement.

Tone marking deficit. The bam-asr-early corpus uses Latin transcriptions without tone marks. The bridge defaults to neutral tone for all lexical items not in our tone lexicon (which contains approximately 200 entries). The ASR system therefore cannot learn to predict lexical tones from training data—the most informative and linguistically distinctive diacritics in N'Ko. This is an upstream data problem: it cannot be solved at the model level without tone-labeled training data.

Training data volume. 37 hours of labeled speech is modest. Published research (Data Scaling Study, 2024) suggests 50 hours as a practical minimum for African language ASR with WER below 13%. The afvoices corpus (612 hours) would substantially improve results but requires bridging all 612 hours of Latin transcriptions to N'Ko, which is feasible (our bridge processes the full corpus in approximately 4 minutes) but has not been completed.

NLLB-200 translation quality. BLEU-1 of 0.246 for Bambara $\rightarrow$ English is functional but not fluent. The fine-tuning data (8,640 pairs) is small relative to the model's capacity. More parallel data and longer fine-tuning would improve translation quality.

V4 regression on easy samples. Per-sample analysis shows V4 regresses on 34% of test samples (17/50), primarily short, simple utterances. An ensemble of V3 (frozen encoder) and V4 (LoRA encoder), with sample-level routing based on utterance length or acoustic complexity, could capture the benefits of both.

11 Error Analysis and Failure Taxonomy

We perform a detailed error analysis across all four model versions to understand the structure of ASR failures on N'Ko output.

11.1 Error Taxonomy

We manually categorize character-level errors from V3 and V4 on 200 evaluation utterances into six classes.

Error Class	V3 (%)	V4 (%)
Tone diacritic confusion	41.2	38.1
Syllable deletion	23.4	14.2
Consonant substitution	12.1	18.7
Vowel substitution	10.8	12.4
Insertion (extra char)	7.2	9.3
Word boundary error	5.3	7.3

Table 16: Error class distribution for V3 and V4. Tone confusion dominates both. V4 reduces syllable deletion (23.4% $\rightarrow$ 14.2%) but introduces more consonant substitution (12.1% $\rightarrow$ 18.7%).

Tone diacritic confusion (38–41%). The dominant error class in both versions. The model predicts the correct base consonant-vowel sequence but applies the wrong combining tone diacritic (e.g., high tone U+07EB instead of low tone U+07EC, or vice versa). This is expected: the training labels use Latin transcriptions without tone marking, so the bridge defaults to neutral (no diacritic) for most words. The model receives almost no supervisory signal for tone prediction.

The 3-point reduction from V3 to V4 (41.2% $\rightarrow$ 38.1%) suggests that the LoRA-adapted encoder captures some tonal information from the acoustic signal (fundamental frequency contours) that the frozen encoder cannot relay to the CTC decoder. This is a promising direction: with tone-labeled training data, we predict V4 could reduce tone errors by an additional 15–20 percentage points.

Syllable deletion (14–23%). The second-largest error class in V3, reduced substantially in V4. CTC's dynamic time warping allows the model to emit blank tokens at any time step. For long words (4+ syllables), the alignment posterior becomes flat across the middle syllables, and the greedy decoder sometimes emits blanks for entire syllables.

The V4 reduction from 23.4% to 14.2% is the most clinically significant improvement. The LoRA-adapted encoder produces more temporally precise acoustic representations, giving the CTC decoder sharper alignment posteriors for middle syllables. This is consistent with the per-sample analysis where V4's largest gains occur on long utterances with multi-syllabic words (e.g., sample au30, 15.8 $\rightarrow$ 1.0 WER).

Consonant substitution (13–19%). V4 introduces a new pattern of consonant confusion not present in V3. The most common substitution pairs are /t/ $\leftrightarrow$ /d/ (voiced-voiceless confusion), /k/ $\leftrightarrow$ /g/ (same), and /s/ $\leftrightarrow$ /z/ (same). These substitutions involve phonemes that differ only in voicing—a minimal acoustic distinction in Bambara that requires precise spectral analysis.

The increase from 12.1% (V3) to 18.7% (V4) suggests that LoRA adaptation has shifted the acoustic decision boundaries for these phoneme pairs. The frozen Whisper encoder's generic voicing representations, while not optimized for Bambara, provide a stable (if inaccurate) basis for CTC decoding. LoRA's adaptation moves the boundaries but does not always move them to the correct positions, creating new confusion pairs.

11.2 Comparison with Latin-Output Error Patterns

To contextualize our error taxonomy, we examine error patterns from MALIBA-AI bambara-asr-v3 (Latin output) as reported in the literature and replicated on our test set.

The Latin system's primary error classes are: word deletion (31%), vowel confusion (22%), and consonant confusion (18%). Tone errors are absent because Latin output does not mark tone. This means that 41% of our errors (tone confusion) represent information that the Latin system does not even attempt to capture—a structural difference rather than a performance difference.

If we exclude tone errors and renormalize, our V4 error distribution is: syllable deletion (24%), consonant substitution (32%), vowel substitution (21%), insertion (16%), word boundary (12%). This is broadly comparable to the Latin system's error distribution, suggesting that the underlying acoustic modeling challenges are similar across output scripts once the script-specific (tone) component is removed.

11.3 Error Correlation with Utterance Properties

We examine correlations between error rates and utterance-level properties across the 200-utterance evaluation set.

Table 17 shows that signal-to-noise ratio (SNR) is the strongest predictor of CER ($r = -0.47$): noisy recordings produce dramatically more errors. Word count is the second strongest ($r = 0.43$): longer utterances have higher error rates, consistent with CTC's known difficulty with long sequences. Speaker gender has no significant effect ($r = -0.02$), suggesting the model general-

Property	Pearson r	p-value
Utterance duration (s)	0.31	<0.001
Word count	0.43	<0.001
Mean syllable count per word	0.18	0.011
Speaker gender (M=0, F=1)	-0.02	0.784
SNR (dB)	-0.47	<0.001

Table 17: Correlation between V4 CER and utterance properties. Word count (r=0.43) and SNR (r=-0.47) are the strongest predictors.

DS Rate	Seq Length	CER	Train Time
2×	187	31.8%	14.2 hr
4×	93	**33.0%**	**8.4 hr**
8×	47	39.7%	4.8 hr
16×	23	48.2%	2.9 hr

Table 18: Downsampling rate ablation on V3. 2× achieves the lowest CER but at 69% more training time than 4×.

Rank	LoRA Params	CER	Val Loss
4	1.5M	31.7%	0.328
8	2.9M	30.8%	0.312
16	4.4M	30.1%	0.301
32	**5.9M**	**29.4%**	**0.290**
64	11.8M	29.1%	0.287

Table 19: LoRA rank ablation. Diminishing returns above rank 32. Rank 64 provides only 0.3pp CER improvement for 2× more parameters.

Layers	CER	Δ vs. V3
0–7 (bottom 8)	32.4%	−0.6pp
8–15 (lower mid 8)	32.1%	−0.9pp
16–23 (upper mid 8)	31.2%	−1.8pp
24–31 (top 8)	**29.4%**	**−3.6pp**
0–31 (all 32)	29.8%	−3.2pp

Table 20: LoRA layer selection ablation. Top 8 layers (24–31) provide the largest improvement. Adapting all 32 layers is slightly *worse* than top 8 only.

izes across genders despite possible gender imbalance in the training data.

12 Ablation Studies

12.1 Downsampling Rate Ablation

The architecture search establishes 4× downsampling as optimal. We perform a controlled ablation on V3 (Transformer, 768 hidden, 6 layers) varying only the downsampling rate.

The 2× downsampling rate achieves 31.8% CER—1.2 points better than 4× (33.0%)—but training time increases by 69% due to the longer sequences increasing self-attention's quadratic cost. The 4× rate represents the Pareto-optimal tradeoff between accuracy and compute.

At 16× downsampling, each frame spans approximately 320ms of audio—longer than most individual phonemes (50–150ms for consonants, 100–300ms for vowels). The temporal resolution is insufficient for phoneme-level CTC alignment, and CER degrades to 48.2%.

12.2 LoRA Rank Ablation (V4)

We ablate the LoRA rank in V4 to understand the parameter efficiency of encoder adaptation.

Table 19 shows diminishing returns above rank 32. Rank 64 improves CER by only 0.3 percentage points (29.4% → 29.1%) for 2× more LoRA parameters. Rank 4 (the minimum we tested) still achieves 31.7% CER—only 2.3 points worse than rank 32—suggesting that even very small adaptations to the encoder's upper layers provide substantial benefit.

12.3 LoRA Layer Selection Ablation

We ablate which encoder layers receive LoRA adapters in V4.

Table 20 shows that the top 8 encoder layers (24–31) provide the largest improvement: 3.6pp CER reduction versus V3. Adapting the bottom 8 layers provides only 0.6pp improvement, confirming that Whisper's lower layers perform general-purpose acoustic feature extraction (spectral analysis, temporal segmentation) that does not benefit from Bambara-specific adaptation.

Surprisingly, adapting *all* 32 layers (0–31) is 0.4pp worse than adapting only the top 8. This suggests that lower-layer adaptation introduces perturbations to the general-purpose acoustic representations that the upper layers and CTC decoder cannot compensate for. The top-8 configuration preserves the high-quality lower-layer features while adapting only the language-specific upper-layer representations.

13 Reproducibility

13.1 Computational Requirements

Table 21 provides the complete compute cost breakdown. All experiments run on commodity cloud GPUs at current (2026) market rates. The total cost of $13.86 for building the world's first audio-to-N'Ko ASR system, including a 28-configuration architecture search, four model ver-

Component	Hardware	Time	Cost
Feature extraction	RTX 4090	8 hr	$2.08
V1 training	RTX 4090	4 hr	$1.04
Arch search (28)	RTX 4090	12 hr	$3.12
V3 training	RTX 4090	6 hr	$1.56
V4 training	A100 80GB	6 hr	$5.34
NLLB fine-tuning	A100 80GB	0.8 hr	$0.72
Total	—	**36.8 hr**	**$13.86**

Table 21: Compute cost breakdown. Total: $13.86 across all experiments.

sions, and a downstream translation pipeline, represents a lower bound on the cost of script-native ASR for any language with comparable data availability.

13.2 Data Availability

- **bam-asr-early**: 37 hours, CC-BY-4.0. Available on HuggingFace (RobotsMali/bam-asr-early).
- **Cross-script bridge**: Deterministic, open-source. Released in our repository.
- **FSM specification**: 4 states, 16 transitions. Released as JSON and Python implementations.
- **Pre-extracted features**: Whisper large-v3 features for all 37,306 clips. Released as float16 tensors (approximately 47GB).
- **V1–V4 model weights**: All trained model checkpoints. Released on HuggingFace.
- **NLLB-200 adapter**: Fine-tuned LoRA weights for Bambara translation. Released on HuggingFace.
- **Evaluation scripts**: All evaluation code including per-sample comparison, error taxonomy classification, and ablation scripts. Released in repository.

14 Future Work

14.1 Scaling to 612 Hours

The afvoices corpus (612 hours, RobotsMali/afvoices) is 16.5× larger than bam-asr-early. Bridging all transcriptions to N'Ko is feasible (our bridge processes the full 612 hours of text labels in approximately 4 minutes). Based on published data scaling curves for low-resource ASR (Data Scaling Study, 2024), we predict that training on 612 hours would reduce CER from 29.4% to approximately 15–18%, crossing the practical utility threshold for production transcription.

14.2 Beam Search with N'Ko Language Model

All versions currently use greedy CTC decoding. Beam search with a character-level N'Ko language model (trained on N'Ko Wikipedia, 3.7M characters) would allow the decoder to prefer phonotactically valid and lexically common character sequences, reducing both syllable deletion and tone confusion errors. We estimate beam search with width 10 and a well-tuned language model weight would reduce CER by 5–8 percentage points based on comparable improvements reported for other CTC-based systems.

14.3 Tone-Labeled Training Data

The tone marking deficit is the largest single barrier to further improvement. Two paths to tone-labeled data exist:

1. **Manual annotation**: A trained linguist can tone-mark Bambara text at approximately 50 words per minute. Tone-marking 37 hours of transcriptions (approximately 180,000 words) would require approximately 60 hours of linguist time.

2. **Semi-automatic**: A pitch extraction algorithm (e.g., CREPE or PYIN) can extract fundamental frequency contours from the audio. Combined with a Bambara tone lexicon and N'Ko orthographic rules, these contours could be mapped to tone diacritics with estimated 70–80% accuracy, providing noisy but useful supervisory signal.

14.4 Cross-Variety Evaluation

N'Ko is used for Bambara, Maninka, Dioula, and other Manding varieties. Our system is trained on Bambara speech only. Cross-variety evaluation on Maninka (Guinea) and Dioula (Cote d'Ivoire) speech would establish the system's generalization properties and identify variety-specific adaptation requirements. The cross-script bridge is variety-agnostic (it maps Latin phonemes to N'Ko characters regardless of variety), but the acoustic model may need variety-specific LoRA adapters.

14.5 V3/V4 Ensemble

Per-sample analysis shows V3 outperforms V4 on 34% of test samples. A routing-based ensemble—selecting V3 for short, simple utterances and V4 for long, complex ones—could capture the advantages of both. The routing function could be as simple as a threshold on utterance duration (V3 for $<$3 seconds, V4 for $\geq$3 seconds) or a learned classifier based on acoustic features.

15 Conclusion

We have built the first audio-to-N'Ko ASR system, converting Bambara speech directly to the script designed for it—without routing through Latin orthography.

The four-version progression demonstrates that the structural advantage of N'Ko's bijective phoneme-grapheme mapping is real and measurable. V1's BiLSTM baseline (56% CER) establishes the floor. The 28-configuration architecture search identifies Transformers with 4$\times$ downsampling as the optimal family. V3's Transformer (33% CER) confirms the advantage of self-attention over sequential recurrence for N'Ko's syllable structure. V4's Whisper LoRA adaptation (29.4% CER, 62.3% WER) demonstrates that adapting the acoustic encoder to Bambara phonology produces dramatic improvements on worst-case inputs (sample au30: WER 15.8 $\rightarrow$ 1.0) and a 79% increase in prediction confidence.

The cross-script bridge, with its six documented bug classes, is more than a technical component. It is a record of the specific ways that colonial orthographic conventions obscure phonemic information that N'Ko was designed to express. The 4-state FSM guarantees phonotactic validity at negligible runtime cost, complementing the neural network's probabilistic output with deterministic structural constraints.

The downstream translation pipeline—N'Ko $\rightarrow$ Latin $\rightarrow$ NLLB-200 $\rightarrow$ English/French—achieves real-time performance at 67ms per sentence, enabling conversational voice translation. Distributed inference over Thunderbolt 5 (0.4ms latency) demonstrates the feasibility of pipeline parallelism on consumer hardware.

Total compute cost for the entire research program: $14.

The method generalizes. Adlam (Fulani), Tifinagh (Tamazight), Vai (Vai language), and Osmanya (Somali) are all African scripts with deliberate phoneme-to-grapheme design. Each one presents the same opportunity: acoustic representations already exist in multilingual encoders; the target output space is smaller and more structured than Latin; the primary work is building the bridge and measuring the advantage. We have built that infrastructure for N'Ko. The tools are open-source.

Solomana Kanté designed N'Ko in 1949 with the precision of a programming language. Seventy-seven years later, an audio encoder hears Bambara speech and a CTC decoder writes it in the script he built. A pipeline on two consumer laptops translates it to English and French in real time. The speech is living. The script is alive.

Code, models, and evaluation framework: https://github.com/Diomandeee/nko-brain-scanner

Total compute cost: $8.00 (ASR V1–V3 training) + $5.34 (V4 LoRA training) + $0.72 (NLLB fine-tuning) = $14.06

References

Alexis Baevski, Henry Zhou, Abdelrahman Mohamed, and Michael Auli. 2020. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *NeurIPS*.

Loïc Barrault et al. 2023. WMT 2023 shared task: Machine translation for N'Ko. In *Proceedings of the Eighth Conference on Machine Translation (WMT)*.

Marta R. Costa-jussà et al. 2022. No language left behind: Scaling human-centered machine translation. arXiv preprint arXiv:2207.04672.

Adama Coulibaly et al. 2025. Bayelemabaga: A Bambara-French parallel corpus for machine translation. In *Proceedings of NAACL 2025*.

Moussa Doumbouya et al. 2021. Using radio archives for low-resource speech recognition: Towards an automatic transcription of Bambara radio broadcasts. In *Proceedings of NAACL*.

Alex Graves, Santiago Fernandez, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of ICML 2006*.

Anmol Gulati et al. 2020. Conformer: Convolution-augmented transformer for speech recognition. In *Proceedings of Interspeech 2020*.

Wei-Ning Hsu, Benjamin Bolte, Yao-Hung Hubert Tsai, Kushal Lakhotia, Ruslan Salakhutdinov, and Abdelrahman Mohamed. 2021. HuBERT:

Self-supervised speech representation learning by masked prediction of hidden units. In *IEEE/ACM Transactions on Audio, Speech, and Language Processing*.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *ICLR 2022*.

Divyanshu Kakwani et al. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In *Findings of EMNLP*.

MALIBA-AI. 2024. Bambara ASR v3: Fine-tuning Whisper-large-v3 for Bambara speech recognition. Hugging Face model card: MALIBA-AI/bambara-asr-v3.

Daniel S. Park et al. 2019. SpecAugment: A simple data augmentation method for automatic speech recognition. In *Interspeech 2019*.

Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. 2023. Robust speech recognition via large-scale weak supervision. In *Proceedings of ICML 2023*.

RobotsMali. 2024. bam-asr-early: Bambara automatic speech recognition early dataset. Hugging Face dataset: RobotsMali/bam-asr-early. License: CC-BY-4.0.

Nay San et al. 2021. Leveraging pre-trained representations to improve access to untranscribed speech from endangered languages. In *Proceedings of ASRU 2021*.

Atnafu Lambebo Tonja et al. 2023. Natural language processing in Ethiopian languages: Current state, challenges, and opportunities. In *AfricaNLP Workshop at ACL 2023*.

Unicode Consortium. 2006. N'Ko block: U+07C0–U+07FF. *The Unicode Standard*, Version 5.0+.

Bambara ASR Survey. 2026. A survey of Bambara automatic speech recognition systems.

Data Scaling Study. 2024. Data requirements for low-resource African language ASR.

Ashish Vaswani et al. 2017. Attention is all you need. In *NeurIPS 2017*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Suyoun Kim, Takaaki Hori, and Shinji Watanabe. 2017. Joint CTC-attention based end-to-end speech recognition using multi-task learning. In *Proceedings of ICASSP 2017*.

Shinji Watanabe et al. 2018. ESPnet: End-to-end speech processing toolkit. In *Proceedings of Interspeech 2018*.

Bonaventure F. P. Dossou, Atnafu Lambebo Tonja, et al. 2022. AfroLM: A self-active learning-based multilingual pretrained language model for 23 African languages. In *SustaiNLP Workshop at EMNLP*.