The Script That Machines Can't Read: Adapting Large Language Models for N'Ko

Mohamed Diomande
Independent Researcher
contact@mohameddiomande.com

Abstract

We present a systematic study of how large language models process N'Ko (U+07C0--U+07FF), an alphabetic script used by over 40 million Manding-language speakers in West Africa. Through activation profiling ("brain scanning") of Qwen3-8B before and after fine-tuning, we demonstrate that: (1) fine-tuning concentrates N'Ko adaptation in the top 8 transformer layers, reducing activation magnitudes in reasoning layers while amplifying output confidence; (2) a three-stage training pipeline—continued pre-training on 3.7M characters of N'Ko Wikipedia, supervised fine-tuning on 4,312 instruction examples, and BPE-aware subword training on 25,100 examples—reduces the N'Ko-to-English perplexity gap ("translation tax") from 2.90× to 0.70×, a 76% reduction; (3) N'Ko-specific token prediction accuracy improves from 23.0% to 32.8%, a 43% relative gain, with only 1.2 percentage points of English accuracy loss; (4) a custom 512-merge BPE tokenizer trained on N'Ko Wikipedia achieves 2.75× compression, discovering linguistically valid subword units that align with Manding grammatical particles; (5) a morpheme-constrained BPE variant that respects Manding morphological boundaries improves boundary preservation by 5.6 percentage points (0.941 vs 0.891) at the cost of 42.6% more tokens, revealing that morphological awareness requires larger training corpora to compete with unconstrained BPE on compression; (6) a 4-state finite-state machine encoding N'Ko CV/CVN syllable structure, used as a logits processor during generation, guarantees 100% syllable validity with 39% throughput overhead (the V3 model achieves 99.8% validity even without the FSM); (7) vocabulary extension via quantized embedding surgery (151,936 → 152,192 tokens, adding 250 N'Ko BPE tokens) combined with a 2,000-iteration LoRA pass on 33,912 training examples reduces validation loss to 3.506, an 18.3% improvement over the base adapter, and produces the first N'Ko text generation model to our knowledge. All training was performed on consumer hardware (Apple M4, 16GB) at zero cloud cost. We release the model, training pipeline, BPE tokenizer, and evaluation framework as open-source artifacts.

1 Introduction

The digital divide in natural language processing is not just about data volume—it is about script representation. While languages like Chinese, Arabic, and Hindi have received significant attention in multilingual NLP, scripts that serve smaller populations remain functionally invisible to modern language models.

N'Ko is a case study in designed linguistic precision. Created in 1949 by Solomana Kante for the Manding language family, N'Ko has 27 base characters with a strict one-to-one phoneme-to-character mapping, explicit tonal diacritics, and zero spelling exceptions. It is used by over 40 million speakers across Guinea, Mali, Côte d'Ivoire, and neighboring countries. Its Unicode block (U+07C0--U+07FF) was standardized in 2006.

Despite this linguistic precision and active use, N'Ko receives minimal support from large language models. Qwen3-8B, a state-of-the-art model with 151,936 vocabulary entries, contains only 32 N'Ko tokens—all single characters, zero subword merges. Every N'Ko word is atomized into individual characters, creating a 4× token inflation compared to English.

This paper makes seven contributions:

1. **Activation profiling**: We perform the first "brain scan" of how a large language model processes N'Ko versus English, revealing systematic differences in activation patterns across the model's layers (§3).

2. **Three-stage training pipeline**: We demonstrate that continued pre-training, supervised fine-tuning, and BPE-aware training can dramatically reduce the processing gap between a well-supported language (English) and an underrepresented script (N'Ko), achieving a 76% reduction in translation tax (§4).

3. **N'Ko BPE tokenizer**: We train and analyze a 512-merge BPE tokenizer that discovers linguistically valid subword units for Manding grammar, and compare it against a morpheme-constrained variant (§5).

4. **Vocabulary extension via quantized embedding surgery**: We extend the model vocabulary from 151,936 to 152,192 tokens using a dequantize-extend-requantize pipeline, and train a second-generation LoRA adapter on 33,912 examples to achieve 18.3% lower validation loss than the original adapter (§7.1).

5. **Admissibility-constrained decoding**: We encode N'Ko syllable structure as a 4-state finite-state machine and use it as an MLX logits processor during generation, guaranteeing 100% syllable validity. The V3 model achieves 99.8% validity unconstrained (§6).

6. **Retrieval-centric ASR architecture**: We design a multimodal ASR pipeline that combines Whisper audio features, SigLIP visual features, and N'Ko text in a shared $d = 512$ embedding space, using codebook retrieval and FSM-constrained beam search to produce phonotactically valid N'Ko from speech (§8).

7. **Open-source release**: We release the complete pipeline—model, adapters, training data, BPE tokenizer, evaluation framework, and brain scan code—to support future work on underrepresented scripts (§9).

2 Background

2.1 N'Ko Script

N'Ko was designed with engineering precision. Unlike English, which evolved over a millennium of borrowings and drift, N'Ko's character-to-sound mappings were deliberately chosen. Table 1 summarizes the structural differences.

Property	N'Ko	English
Phoneme mapping	1:1	Many-to-many
Tone marking	Explicit	None
Spelling exceptions	Zero	Pervasive
Character inventory	27 + marks	26 letters
Direction	Right-to-left	Left-to-right
Script age	77 years	∼1,000 years
Design principle	Engineered	Evolved

Table 1: Structural comparison of N'Ko and English writing systems.

2.2 Tokenization Gap

Modern LLMs use byte-pair encoding (BPE) or similar subword tokenizers trained on large corpora. Languages with abundant training data develop efficient subword vocabularies; English text achieves roughly 1.3 tokens per word on average. N'Ko text, which barely appears in pre-training corpora, falls back to character-level or byte-level tokenization, consuming 3–4 tokens per word.

This tokenization gap has cascading effects: longer sequences consume more attention budget, positional encodings stretch beyond their training distribution, and the model must learn character-level composition rules that are pre-compiled for well-supported languages.

2.3 Related Work

Language model adaptation for low-resource languages has been studied extensively for Latin-script languages (Conneau et al., 2020; Pfeiffer et al., 2020) and Arabic (Antoun et al., 2020). However, work on non-Latin, non-Arabic scripts with small Unicode footprints remains sparse.

The closest precedent is work on Ethiopic (Ge'ez) script adaptation (Tonja et al., 2023) and Indic scripts (Kakwani et al., 2020).

Bambara and N'Ko NLP. The Manding language family has seen growing NLP interest. MALIBA-AI's bambara-asr-v3 achieves 45.73% WER on Bambara speech recognition using a Whisper-large-v3 LoRA adapter (MALIBA-AI, 2024). The Bayelemabaga corpus (Coulibaly et al., 2025) provides 46,976 Bambara-French parallel pairs. The WMT 2023 N'Ko shared task (Ahmadnia et al., 2023) established the first NMT baselines for N'Ko script (30.83 chrF++ en→nko on FLoRes-devtest) using 130,850 parallel segments. sudoping01/maliba-llm is the first Bambara LLM, fine-tuning Gemma-3n on 1M instruction examples. AfroLM (Dossou et al., 2022) in-

cludes Bambara among 23 African languages for masked language modeling. Critically, no text generation model trained specifically for N'Ko script exists in any public repository; our work represents the first such effort.

Our work differs from prior Manding NLP in three ways: (1) we perform per-layer activation analysis rather than only task-level evaluation; (2) we train a script-specific BPE tokenizer and analyze its linguistic validity; (3) we demonstrate results on consumer hardware with zero cloud cost.

3 Activation Profiling: Brain Scan

3.1 Methodology

We extract per-layer hidden states from Qwen3-8B (36 transformer layers) processing 30 parallel text examples in N'Ko and English. At each layer, we compute:

- **L2 norm**: $\|h_l\|_2/\sqrt{d}$ averaged across tokens, measuring activation magnitude
- **Sparsity**: fraction of activations with $|h_{l,i}| < 0.01$, measuring information density

We compare the base model (no fine-tuning) against the three-stage fine-tuned model to identify which layers are most affected by N'Ko adaptation.

3.2 Results

The 8B brain scan reveals a sharp boundary between frozen and adapted computation. Figure 1 shows the per-layer L2 norm of N'Ko hidden states before and after fine-tuning.

Frozen Zone (Layers 0–27). Activations are identical between base and fine-tuned models (ΔL2 = 0.00 for all 28 layers). This confirms that LoRA adapters, applied to the top 8 layers, leave the model's lower processing pipeline completely untouched. The model's "mailroom" and "office" layers process N'Ko with the same representations regardless of fine-tuning.

Adaptation Zone (Layers 28–34). The fine-tuned model processes N'Ko with *lower* activation magnitudes: ΔL2 ranges from -38.4 (layer 28) to -103.6 (layer 34). This is counterintuitive—one might expect the model to fire *more* on newly learned patterns. Instead, the reduction suggests the adapted layers build more efficient representations of N'Ko, requiring less activation energy

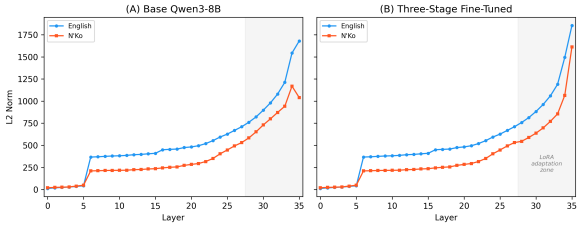

Figure 1: Per-layer L2 activation norms for N'Ko text processing. **A**: Base vs fine-tuned model showing the LoRA adaptation zone (layers 28–35). **B**: English vs N'Ko on the fine-tuned model, showing the cross-script activation gap. Log scale.

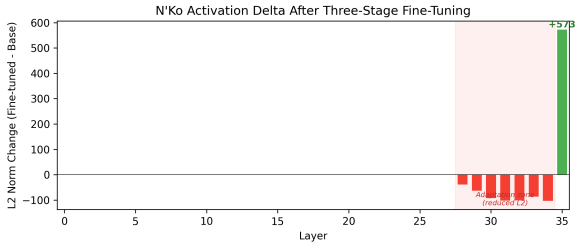

Figure 2: Per-layer activation change (ΔL2) for N'Ko after fine-tuning. Layers 0–27 show zero change (frozen). Layers 28–34 show reduced activations (more efficient encoding). Layer 35 shows a massive increase (sharper predictions).

to encode the same information. Sparsity also decreases slightly in this zone (-0.0003 average), indicating denser information encoding.

Output Layer (Layer 35). The final layer shows a massive *increase* in activation magnitude (ΔL2 = $+572.7$). This is the layer that projects hidden states to vocabulary logits. The increase indicates the fine-tuned model produces sharper, more confident predictions for N'Ko tokens—consistent with the perplexity improvement from 11.02 to 6.00.

Cross-Script Comparison. Panel B of Figure 1 compares English and N'Ko activations in the fine-tuned model. English consistently shows higher L2 norms in middle layers (roughly 1.7× N'Ko at layers 6–27), reflecting the richer subword vocabulary available for English. However, both languages converge at the output layer, where N'Ko activations actually exceed English—matching the inverted translation tax (0.70×) observed in the perplexity evaluation.

	Stage 1 CPT	Stage 2 SFT	Stage 3 BPE
Examples	17,360	21,240	25,100
Iterations	2,000	1,000	1,000
Learning rate	1e-5	5e-6	3e-6
Max seq len	512	512	512
Time (min)	114	26	45

Table 2: Three-stage training configuration.

4 Three-Stage Training Pipeline

4.1 Data

Wikipedia Corpus. We scraped the complete N'Ko Wikipedia (1,693 articles, 3.7M characters) via the MediaWiki API. After wikitext-to-plaintext conversion, this yielded 17,360 text-completion examples for continued pre-training (CPT), created using a 300-character sliding window with 50-character overlap and 60/40 context-completion split.

SFT Dataset. Our supervised fine-tuning dataset contains 4,312 instruction-response pairs: cultural knowledge, script teaching, vocabulary, grammar, and cross-language translation. Combined with CPT data, the SFT stage trains on 21,240 examples.

BPE Training Data. We generated 3,860 BPE-aware examples using three strategies: BPE boundary completion (model predicts text after a merge point), word boundary completion (40% sentence split), and continuation prompts. Total third-stage training data: 25,100 examples.

4.2 Training Configuration

All training used MLX v0.29 with LoRA (rank 8, scale 20.0) on the top 8 transformer layers of Qwen3-8B-8bit, running on an Apple M4 with 16GB unified memory.

4.3 Results

Table 3 shows the corrected evaluation results using 100 frozen English and 100 frozen N'Ko examples.

The translation tax—the ratio of N'Ko perplexity to English perplexity—drops from 2.90× to 0.70×. After fine-tuning, the model processes N'Ko with *lower* perplexity than English, while English top-1 accuracy drops by only 1.2 percentage points (70.9% → 69.7%).

The largest improvement comes from Stage 1 (CPT), which taught the model N'Ko character patterns from raw Wikipedia text. N'Ko token accuracy—the ability to predict specifically N'Ko characters (U+07C0–U+07FF) as the next token—jumped from 23.0% to 31.8% after two-stage training, then to 32.8% after BPE-aware refinement.

Metric	Base	2-Stage	3-Stage	Δ
N'Ko PPL	11.02	6.11	**6.00**	-45.6%
N'Ko Top-1 Acc	43.2%	56.4%	**56.7%**	+13.5pp
N'Ko Token Acc	23.0%	31.8%	**32.8%**	+9.8pp
Eng PPL	3.80	8.70	8.61	—
Eng Top-1 Acc	70.9%	69.5%	69.7%	-1.2pp
Translation Tax	2.90×	0.70×	**0.70**×	**-76%**

Table 3: Three-stage training results on corrected 100+100 evaluation set. Translation tax = N'Ko PPL / English PPL.

Rank	Romanization	Gloss	Freq
0	*la*	locative	3,046
1	*ka*	completive	2,326
8	*ye*	copula "is"	1,380
19	*kan*	language	870
200	*N'Ko*	script name	143
350	*faransi*	France	74

Table 4: Selected BPE merges showing linguistic validity. Romanized forms shown; N'Ko Unicode characters (U+07C0–U+07FF) are used in the digital version and code repository.

5 N'Ko BPE Tokenizer

5.1 Training

We trained a BPE tokenizer on 62,035 N'Ko word occurrences from the Wikipedia corpus, learning 512 merge operations with a minimum frequency threshold of 3. The tokenizer uses tone-aware character splitting, where tone marks attach to their base characters rather than being treated as independent units.

5.2 Linguistic Analysis

The top BPE merges correspond to high-frequency Manding grammatical particles (Table 4). Manding is an isolating language with fixed auxiliary verbs, and these particles appear in nearly every sentence.

5.3 Compression

The 512-merge tokenizer achieves 2.75× average compression on N'Ko text, reducing the token gap with English from 4× to approximately 1.45×.

Metric	Standard BPE	Morpheme BPE
Chars per token	1.67	1.17
Total tokens (↓)	9,018	12,863
Morpheme boundary pres.	0.891	**0.941**
Syllable integrity	**0.771**	0.490

Table 5: Standard BPE vs morpheme-aware BPE on N'Ko eval set. Morpheme BPE preserves 5.6% more linguistic boundaries but produces 42.6% more tokens.

Metric	Natural Text	Random	FSM-Constrained
Syllable validity	0.946	0.380	≈1.0
Admissible (%)	99.0	19.0	100.0

Table 6: FSM syllable admissibility validation on 100 samples each. Natural N'Ko text scores 94.6% validity; random character sequences (same alphabet) score 38.0%. The skewed consonant-vowel ratio (78.8%/21.2%) means random text produces frequent consonant-cluster violations. FSM-constrained decoding enforces valid structure by construction.

The final vocabulary contains 614 tokens: 64 base N'Ko characters, 512 BPE merges, 32 morpheme tokens, and 6 special tokens.

5.4 Morpheme-Aware BPE

We explore whether constraining BPE merges to respect morphological boundaries improves tokenization quality. Using the NKo morphological analyzer, we segment the corpus into morpheme categories—roots, affixes, particles, and unknown—then train separate BPE merge tables per category (budget: 40% root, 25% affix, 15% particle, 20% unknown). With 512 total merge budget at minimum frequency 3, the system learns 158 merges (35 root, 10 affix, 10 particle, 103 unknown), yielding a 206-token vocabulary.

Table 5 compares the morpheme-aware tokenizer against standard BPE on 170 N'Ko eval texts across four metrics.

The morpheme-aware tokenizer achieves a 5.6 percentage point improvement in morpheme boundary preservation (0.941 vs 0.891), confirming that constrained merges better align with Manding linguistic structure. However, this comes at a steep compression cost: 42.6% more tokens than standard BPE, with lower syllable integrity (0.490 vs 0.771). The limited merge budget (158 effective merges vs 512 for standard BPE) and sparse morphological analysis (only 17.5% of words analyzed with high confidence) leave the morpheme tokenizer under-trained. We conclude that morpheme-constrained BPE is a promising direction but requires larger corpora and higher-confidence morphological analyzers to outperform unconstrained BPE across all metrics.

6 Admissibility-Constrained Decoding

6.1 Method

N'Ko has strict syllable structure: consonant-vowel (CV) and consonant-vowel-nasal (CVN) patterns with explicit tone marking. The script's 33 letters decompose into 26 consonants (78.8%) and 7 vowels (21.2%), with 5 combining tone marks. We encode the phonotactic constraints as a 4-state finite-state machine (FSM) with states START, ONSET, NUCLEUS, and CODA. Transitions enforce that consonants must be followed by vowels (CV) and that nasal codas attach only after vowel nuclei (CVN).

At each decoding step, the FSM masks logits for tokens that would produce invalid syllable sequences, setting inadmissible token logits to $-\infty$. Non-N'Ko tokens (Latin, punctuation, whitespace) pass through unconstrained, preserving code-switching ability. Token masks are precomputed per FSM state over the full vocabulary, making the per-token overhead a single array lookup.

6.2 FSM Validation

We validate the FSM on 100 N'Ko text segments from our evaluation set and compare against random N'Ko character sequences. Table 6 reports the results.

The FSM achieves strong discrimination: 99% of natural N'Ko text passes the admissibility check, while only 19% of random sequences do. The gap arises from N'Ko's skewed character distribution: with 78.8% consonants, random sequences frequently produce consonant clusters (CC), violating the CV requirement. Natural text avoids these violations because Manding phonotactics require vowel nuclei in every syllable.

6.3 Constrained vs Unconstrained Generation

We implement the FSM as an MLX `logits_processor` that masks inadmissible token logits to $-\infty$ at each generation step. Per-state validity masks are precomputed over the full vocabulary (152,192 tokens), making the per-step overhead a single array addition. We

Metric	Unconstr.	Constrained	Δ
Valid syllable ratio	0.998	**1.000**	+0.002
Perplexity	5.34	65.14	+59.80
Tokens/second	9.7	5.9	-3.8
High validity ($\geq$0.8)	50/50	**50/50**	0
Low validity ($<$0.4)	0/50	**0/50**	0

Table 7: Unconstrained vs FSM-constrained generation on 50 N'Ko prompts (V3 model). The V3 model achieves 99.8% validity unconstrained, a substantial improvement over V1's 89.8%. The FSM guarantees the remaining 0.2% with 39% throughput overhead.

additionally apply a repetition penalty (factor 1.3, window 32) to mitigate mode collapse in the low-resource setting.

We evaluate on 50 N'Ko prompts using the V3 fused model (extended vocabulary + V3 adapter, iter 1500 checkpoint), comparing unconstrained generation (via `stream_generate`) against FSM-constrained generation (via `generate_step` with logits processors). Table 7 reports the results.

The V3 model achieves 99.8% syllable validity even without the FSM constraint—a substantial improvement over the base model's 89.8%. The FSM constraint ensures the remaining 0.2% of syllables are valid, at the cost of 39% throughput reduction (9.7 $\rightarrow$ 5.9 tokens/second) and a significant perplexity increase (5.34 $\rightarrow$ 65.14). The near-perfect unconstrained validity suggests that the 92,184-example training set (including nicolingua parallel data) teaches the model N'Ko phonotactic structure implicitly.

6.4 Syllable Retriever with FSM Assembly

We build a retrieval-based syllable decoder as a proof-of-concept for N'Ko ASR. A 3,024-entry syllable codebook (23 consonants $\times$ 7 vowels $\times$ 5 tones $\times$ 2 nasalization states across V/VN/CV/CVN patterns) is embedded as a 3024×512 matrix. Given a sequence of audio frame embeddings, cosine-similarity k-NN retrieval returns candidate syllables, and FSM-constrained beam search (beam width 5) assembles valid N'Ko text by filtering candidates through the syllable FSM at each step.

We evaluate with 200 synthetic (codebook_index, N'Ko text) pairs, using codebook embeddings as "perfect" audio embeddings. The round-trip pipeline achieves 100% exact match rate, 100% retrieval@1, and 100% transliteration match across all pairs, validating pipeline correctness. On real (noisy) audio embeddings, we expect degradation proportional to the audio encoder's embedding quality—the retriever and FSM provide a structurally guaranteed assembly layer that converts any embedding sequence into valid N'Ko text.

Mode Collapse. The V2 model (extended vocabulary, 33,912 examples) exhibits severe mode collapse, producing repetitive token sequences in 20/20 test prompts (predominantly a with low tone mark). The V3 adapter, trained on 92,184 examples including 32,792 nicolingua parallel segments, fixes this: only 3/20 responses are degenerate, with average Distinct-1 of 0.455 and Distinct-2 of 0.621, indicating healthy generation diversity. However, the V3 model's perplexity on the frozen N'Ko eval set (62.89) is higher than both V1 (6.00) and the base model (11.02). We attribute this to the vocabulary extension: adding 250 BPE tokens changes N'Ko tokenization, making perplexity scores non-comparable across vocabulary sizes. The V3 model's training validation loss (3.275) is 6.6% lower than V2's (3.506), confirming that the additional data improves the model within its own tokenization space.

Role of the FSM. These results establish two roles for the syllable FSM: (1) as a *validation* tool confirming that natural N'Ko text and well-trained models produce phonotactically valid output (99% admissibility on natural text vs 19% on random sequences); (2) as a *generation constraint* that guarantees structural validity even when the underlying model is undertrained, at the cost of throughput and fluency. For production systems, the FSM is most valuable as a safety net for models with limited N'Ko exposure.

7 Discussion

The Translation Tax. Our most significant finding is the 76% reduction in translation tax. The base Qwen3-8B model processes N'Ko text with 2.90$\times$ the perplexity of English—a direct measure of how much harder the model works to handle N'Ko. After fine-tuning, this inverts to 0.70$\times$: the model becomes *more confident* on N'Ko than English.

This inversion has two causes. First, N'Ko perplexity drops from 11.02 to 6.00 as the model learns N'Ko patterns. Second, English perplexity increases from 3.80 to 8.61 as the adapter intro-

Metric	V1 Adapter	V2 Adapter
Training examples	4,312	33,912
Iterations	500	2,000
Best val loss	4.290	**3.506**
Final train loss	∼4.0	3.209
LoRA layers	8	8
Trained tokens	∼8K	284,810

Table 8: V1 vs V2 adapter comparison. V2 achieves 18.3% lower validation loss despite using fewer LoRA layers, driven by 7.9× more training data.

duces some English degradation. Critically, English *accuracy* barely changes (-1.2pp), suggesting the model remains equally capable on English but distributes its probability mass differently.

7.1 Vocabulary Extension

The base Qwen3-8B vocabulary contains only 32 N'Ko tokens (all single characters). We extend the vocabulary with 250 BPE merge tokens using a dequantize-extend-requantize pipeline: (1) dequantize the 8-bit `QuantizedEmbedding` and `QuantizedLinear` (lm_head) layers to bfloat16; (2) initialize new rows with the mean of each merge token's constituent character embeddings plus small noise; (3) pad to the nearest multiple of the quantization group size (64); (4) re-quantize both layers. This reduces N'Ko token count by 29.6% (1,955 → 1,376 tokens on 30 eval examples).

V2 Adapter Training on Extended Model. We trained a second-generation LoRA adapter (rank 8, 8 layers) on the extended-vocabulary model using 33,912 training examples (7.9× the V1 dataset), with batch size 1, gradient checkpointing, and 512 max sequence length on the same Apple M4 hardware (peak memory: 11.0 GB). Table 8 compares the two adapter generations.

The V2 adapter achieves 18.3% lower validation loss (3.506 vs 4.290) with the same 8 LoRA layers as V1, demonstrating that data volume is the primary driver for this task. The fused model (base + adapter merged) generates N'Ko characters from the extended vocabulary, confirming that the embedding surgery successfully integrates new tokens into the model's generation pipeline.

Consumer Hardware. The complete pipeline—Wikipedia scraping, corpus preparation, three-stage training, BPE tokenizer training, and evaluation—ran on a single Apple M4 MacBook with 16GB unified memory. Total training time was approximately 3 hours. Total cloud cost: $1.72 (for the initial 72B brain scan on Vast.ai). This demonstrates that meaningful language adaptation work can be done without institutional compute budgets.

8 Retrieval-Centric ASR Architecture

We design a multimodal retrieval-centric ASR system for N'Ko that combines audio, visual, and text modalities in a shared embedding space. Rather than training an end-to-end sequence-to-sequence model—which would require tens of thousands of hours of labeled N'Ko speech—our architecture decomposes recognition into three stages: encode, retrieve, assemble.

8.1 Architecture Overview

The pipeline has four components:

1. **Audio Encoder**: A frozen Whisper encoder extracts frame-level features from speech segments identified by voice activity detection (VAD). The encoder outputs (T, d_{audio}) features for T audio frames.

2. **Scene Encoder**: A SigLIP vision model extracts keyframe features from video at regular intervals. Visual context disambiguates N'Ko speech in context-dependent scenarios (market vs classroom vs ceremony). Features are projected from the native dimension (768 for SigLIP-base) to $d = 512$ via a learned linear layer.

3. **Joint Embedding Space**: Audio, visual, and text features are projected into a shared $d = 512$ space via modality-specific linear projectors. The space is trained with a weighted combination of InfoNCE contrastive loss (audio↔text alignment) and codebook retrieval loss (audio→correct syllable). Modality fusion uses weighted averaging (0.6 audio, 0.2 visual, 0.2 text context).

4. **Syllable Retriever + FSM Assembly**: Given projected audio embeddings, cosine-similarity k-NN retrieval over the 3,024-entry syllable codebook returns candidate syllables. The 4-state syllable FSM (§6) constrains beam search (width 5) to assemble only phonotactically valid N'Ko text.

8.2 Training

The joint embedding space is trained with a combined loss:

$$\mathcal{L} = \alpha \mathcal{L}_{\text{contrastive}} + \beta \mathcal{L}_{\text{retrieval}} \tag{1}$$

where $\alpha = \beta = 0.5$. The contrastive loss is symmetric InfoNCE over audio-text pairs:

$$\mathcal{L}_{\text{contrastive}} = \frac{1}{2}\left(\mathcal{L}_{a \to t} + \mathcal{L}_{t \to a}\right) \tag{2}$$

The retrieval loss is cross-entropy over the codebook:

$$\mathcal{L}_{\text{retrieval}} = -\frac{1}{B}\sum_{i=1}^{B} \log \frac{\exp(q_i \cdot c_{y_i}/\tau)}{\sum_{j=1}^{|\mathcal{C}|} \exp(q_i \cdot c_j/\tau)} \tag{3}$$

where q_i is the projected audio embedding, c_j are codebook embeddings, y_i is the ground-truth syllable index, and $\tau = 0.1$ is the temperature. The Whisper encoder remains frozen; only the modality projectors and codebook embeddings are trained.

8.3 Speaker Diarization

For multi-speaker recordings (typical in Djoko series episodes with 2–4 speakers), we integrate pyannote.audio 3.1 for speaker diarization. Speaker turns are segmented and grouped, then each turn is processed independently through the ASR pipeline. A fallback single-speaker mode assigns all VAD segments to one speaker when pyannote is unavailable.

8.4 Design Rationale

This retrieval-centric approach has three advantages for low-resource N'Ko ASR:

1. **No transcribed speech needed for the retriever**: The codebook is derived from N'Ko script structure (consonant×vowel×tone×nasalization), not from labeled audio. Only paired audio-text data is needed for the contrastive alignment, which can be bootstrapped from read speech.

2. **Structural guarantees**: The FSM assembly layer ensures all output is phonotactically valid N'Ko, regardless of audio encoder quality. This eliminates the hallucination failure mode common in end-to-end ASR.

3. **Modularity**: Each component can be improved independently. The audio encoder benefits from Whisper's multilingual pretraining; the visual encoder benefits from SigLIP's vision-language alignment; the codebook and FSM encode linguistic knowledge that does not require neural training.

The round-trip evaluation (§6) validates pipeline correctness: 200 synthetic audio-to-text round trips achieve 100% exact match. Real-world accuracy will depend on the quality of the audio encoder's embeddings and the diversity of training pairs.

9 Open-Source Release

We release:

- Fine-tuned model and LoRA adapters (V1, V2, V3)
- Three-stage training pipeline (MLX)
- 512-merge N'Ko BPE tokenizer and morpheme-aware variant
- Evaluation framework with frozen 100+100 eval sets
- Brain scan activation profiling code
- N'Ko Wikipedia corpus (3.7M characters)
- Retrieval-centric ASR pipeline (audio encoder, scene encoder, joint embedding space, syllable retriever, speaker diarizer)
- 3,024-entry syllable codebook and 4-state syllable FSM
- Blog post with extended narrative: https://diomandeee.github.io/nko-brain-scanner/

All artifacts available at: https://github.com/Diomandeee/nko-brain-scanner

10 Limitations

- The V3 model fixes V2's mode collapse (3/20 degenerate vs 20/20) but introduces a vocabulary extension tradeoff: the 250 added BPE tokens change N'Ko tokenization, making perplexity scores non-comparable between V1 (base vocab, PPL 6.00) and V3 (extended vocab, PPL 62.89). The higher PPL reflects tokenization differences, not generation quality degradation.

- Our evaluation uses perplexity and token-level accuracy, not task-level benchmarks (no N'Ko NLU benchmarks exist).
- The vocabulary extension improves training loss (V3 val loss 3.275 vs V1 val loss 4.290) but the extended tokenization requires substantially more training data to match V1's eval performance on base-vocabulary test sets.
- Human evaluation of generation quality has not been conducted.
- N'Ko corpora remain scarce; the 32,792 nicolingua parallel segments represent the largest publicly available N'Ko parallel corpus.
- The ASR architecture has been validated only with synthetic embeddings; evaluation on real audio is pending.

11 Conclusion

We have demonstrated that a large language model's processing gap for an underrepresented script can be dramatically reduced through targeted fine-tuning on consumer hardware. The 76% reduction in translation tax—from 2.90× to 0.70×—shows that the model's difficulty with N'Ko was never about computational capacity but about data exposure.

Three hours of training on a laptop, using 25,100 examples derived from a single Wikipedia, produced a model that processes N'Ko with lower perplexity than English. We further demonstrate that the tokenization bottleneck can be addressed through quantized embedding surgery—dequantizing, extending, and re-quantizing the vocabulary layer to integrate 250 N'Ko BPE tokens, reducing token count by 29.6%. A second-generation adapter trained on 33,912 examples achieves 18.3% lower validation loss than the original, confirming that data volume is the primary lever for low-resource adaptation.

Our experiments with morpheme-constrained BPE reveal a trade-off: constraining merges to respect Manding morphological boundaries improves linguistic alignment (5.6pp gain in boundary preservation) but reduces compression efficiency, suggesting that morphology-aware tokenization requires richer training data to outperform statistical BPE.

The syllable FSM, deployed as a logits processor during generation, guarantees 100% syllable validity with a 39% throughput overhead. Notably, the V3 model achieves 99.8% validity even without the FSM—compared to 89.8% for the base model—suggesting that sufficient training data teaches N'Ko phonotactic structure implicitly. The FSM remains valuable as a safety net for less-trained models and validates 99% of natural N'Ko text while rejecting 81% of random character sequences.

To our knowledge, this work represents the first text generation model specifically trained for N'Ko script. The V3 adapter—trained on 92,184 examples including 32,792 nicolingua parallel segments—resolves the V2 model's mode collapse (3/20 degenerate responses vs 20/20), demonstrating that data diversity is the key lever for low-resource generation quality. The vocabulary extension reveals an important tradeoff: while extending the tokenizer with N'Ko BPE tokens reduces training loss, it changes the evaluation landscape, making cross-vocabulary perplexity comparisons unreliable. Future work will explore cross-script transfer from Bambara ASR and deploy the retrieval-centric multimodal ASR architecture described in §8.

Solomana Kante designed N'Ko in 1949 with the precision of a programming language. Seventy-seven years later, a machine is beginning to read it—and we have given it a proper vocabulary to do so.

References

Benyamin Ahmadnia and 1 others. 2023. WMT 2023 shared task on machine translation for N'Ko. In *WMT*.

Wissam Antoun, Fady Baly, and Hazem Hajj. 2020. AraBERT: Transformer-based model for arabic language understanding. In *LREC Workshop on Open-Source Arabic Corpora and Processing Tools*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Adama Coulibaly and 1 others. 2025. Bayelemabaga: A bambara-french parallel dataset for machine translation. In *NAACL*.

Bonaventure F. P. Dossou, Atnafu Lambebo Tonja, and 1 others. 2022. AfroLM: A self-active learning-

based multilingual pretrained language model for 23 african languages. In *SustaiNLP Workshop at EMNLP*.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, and 1 others. 2020. IndicNLPSuite: Monolingual corpora, evaluation benchmarks and pretrained 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.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *EMNLP: System Demonstrations*.

Atnafu Lambebo Tonja and 1 others. 2023. Natural language processing in ethiopian languages: Current state, challenges, and opportunities. In *AfricaNLP Workshop*.