Teaching AI to Count in Bantu Languages

A plain-language report on the BantuNomics fine-tuning results

3MegaLabs · BantuNomics · August 2026

The problem, in one paragraph

Today's AI models speak English brilliantly and Bantu languages poorly. Ask a leading AI to write "153" in Kinyarwanda, or to say "seven divided by two equals three, remainder one" in Chitonga, and it produces something that *looks* fluent — right shape, real words — but is simply **wrong**. Because it looks fluent, most tests don't catch it. BantuNomics exists to fix this, by producing carefully verified language data with native speakers. This report explains how we proved, with numbers, that our data works.

What we did

We took an AI model that scores essentially **zero** on Bantu number tasks, taught it with our data for about an hour on a rented computer (cost: a few dollars), and then gave it an exam it had never seen.

Three things make the exam trustworthy:

The model never saw the test questions. The exam questions were locked away

(digitally fingerprinted, so nobody can quietly swap them) before any teaching began, and we checked — automatically — that no test question ever leaked into the lessons.

Grading is all-or-nothing. The answer must be *exactly* right, every word, every

letter. "Almost right" scores zero. This is the harshest possible way to grade, and we chose it deliberately.

We ran a "scrambled textbook" test. We taught an identical copy of the model with

a fake version of our data — same pages, same length, but with all the answers shuffled onto the wrong questions. If teaching *anything* made the model look better, this fake textbook would show it.

What happened

Student	Score on the unseen exam (400 questions)
Untaught model	**0%** — zero correct
Model taught with the **scrambled** textbook	**0%** — zero correct
Model taught with the **real BantuNomics data**	**76%** — 303 correct

The scrambled-textbook result is the heart of the proof. That model actually *learned the style* — its answers look like Bantu number phrases — and it still got **nothing** right. Looking fluent is worthless. The entire improvement, zero to

76%, comes from one thing only: **the content of our data being correct.**

Some details worth savoring:

On **numbers** (write "88" in Chitonga), the taught model scored **95%**.

On **arithmetic** ("9 × 17 = 153" as a full Kinyarwanda sentence), it scored **87%** —

and these are long phrases about numbers it never saw in training. It didn't memorize; it **learned the system**, the way a child who learns to count can count anything.

An earlier round with a smaller dataset scored 57%; enriching the data's coverage

raised it to 76% on a *harder* exam. **The quality of the data is the dial.**

The surprise: even "African-trained" AI starts at zero

We also tested a larger AI model that was specially pre-trained on **twenty African languages** by a university research group. Before any teaching with our data, its score on the same exam: **0%.** It answered with math-homework boilerplate and never attempted a Bantu phrase. After teaching it with our data, it reached **16.5%** — real learning, including the best calendar score of any model we tested — but far below the 76% our best student reached from the same lessons. Two morals: general exposure to African languages does not teach a model to count in Bantu languages, and **which model you teach matters enormously** — a lesson any AI lab evaluating data will recognize.

What it cost

Everything in this report — all the teaching runs, all the exams, including our mistakes — cost roughly **fifty dollars** of rented computer time. The successful experiment at the heart of it cost about **five dollars**. The expensive ingredient is not the computing. It is the verified data — the thing BantuNomics makes.

What went wrong (we keep honest books)

Not everything worked first try, and we log every failure:

One test initially scored a well-taught model **zero** because of a quirk in how that

model formats answers. We caught it because we always read actual answer samples, not just scores — and fixed the test, not the number.

One training run silently used an old copy of the data because of a caching quirk. We

caught it because the arithmetic of the run didn't add up (it finished far too fast), and rebuilt the safeguard so it cannot recur.

Two long runs hit a time limit and were lost; we now budget run time before launching

and save progress along the way.

Every one of these is documented, with its fix, in the technical report. We publish the failures because for a serious buyer, **the honesty is the credibility**.

Why this matters commercially

For AI companies, this demonstration converts a promise into a measurement:

> *"Our verified Bantu data took a model from 0% to 76% exact-match on unseen tasks for
>
> about $5 of computing — and a matched control proves the entire gain came from the
>
> data's content, not the training procedure."*

The same method extends beyond numbers: the BantuNomics ecosystem produces verified data for sounds, words, grammar, health language, and more, across (today) ten live Bantu languages with more arriving weekly — each with native speakers confirming every form, and every claim testable by machine.

What's next

Repeat the headline run three times with different random starts, to publish an

average with error bars (standard scientific practice).

Grade the winner on the full 2,773-question exam (we used a representative 400 for

speed; the full set is locked and waiting).

Extend the same teach-and-verify loop to speech: the same data now exports

English-and-Bantu paired sentences designed for teaching AI to *hear* numbers, not just write them.

Small glossary

Model — the AI program being taught.
Fine-tuning — additional teaching of an existing AI using new material.
Held-out exam — test questions the model has never seen, locked before teaching.
Exact match — grading where only a perfectly correct answer counts.
Control — the scrambled-textbook twin experiment that isolates *why* scores improved.
Baseline — the model's score before any teaching.

Companion to the full technical report (same folder), which contains every number, method, command, and failure in detail.