LOW-RESOURCE LANGUAGE MODEL WITH CYBER DEFENSE

Preprint · January 2025

DOI: 10.13140/RG.2.2.22096.42245

1 author:

Josiah Umezurike
Lokdon
4 PUBLICATIONS **0** CITATIONS

SEE PROFILE

LOW-RESOURCE LANGUAGE MODEL WITH CYBER DEFENSE

Josiah Umezurike
Each1Teach1 Tech -Emerging Tech Research
Columbia, SC, USA
jumezurike@each1teach1.us

ABSTRACT

This paper explores the unique phonemes and morphemes structure of the Igbo, Yoruba, Hausa, and AAVE(Gulah/Gichee) language, focusing on the invariant nature of root words, vowel linking, tones, and their semantic enhancement through affixation. It introduces a unified alphabet (Σ_U) for Igbo, Yoruba, Hausa and African American Vernacular English (AAVE). Nonethnic language models, often based on subject-verb-object structures typical of English, fall short in accurately representing Igbo, Yoruba, Hausa, and AAVE. Hence, we propose a unified alphabet and deep learning strategy for low-resource languages using neural networks. Here, vowels act as activators to capture the phonological and morphological patterns of the language. The paper introduces a method that borrows from the AFA oracle of knowledge equivalent to the Turing machine. This model will also leverage a shift-left approach or security-by-design principles to ensure robust performance and secure data handling. The need for autonomy makes it relevant to incorporate keyless encryption methods, which are resilient against quantum computing threats, guaranteeing the security and privacy of data, transactions, and communications within the model. Machine learning (supervised, unsupervised, and a mix of the two), reinforced by human feedback of a set of language strings over a unified alphabet, comprised the algorithm: Including a dynamic AFA matrix that can represent any conceivable word or phrase in Igbo, Yoruba, Hausa and AAVE, significantly enhancing AI tasks such as translation, text generation, and speech recognition.

***K*eywords** Cybersecurity · Keyless Cryptography · Artificial Intelligence · Language Modeling · Data Security · Post-Quantum Cryptography · Natural Language Processing · Machine Learning.

1 Introduction

In recent years, research on the Large Language Model (LLM) has catalyzed significant innovations in artificial intelligence, particularly within the realms of Machine Translation (MT), Natural Language Processing (NLP) and Retrieval-Augmented Generation (RAG) [1, 2]. These advancements have permeated computational linguistics at large. However, the application of LLMs in low-resource languages presents formidable challenges. The necessity for extensive data sources renders it nearly insurmountable for infrequently used languages to attain the breakthroughs observed in high-resource languages. Challenges such as hallucination, back translation issues, and the absence of cultural nuances and authenticity have led to inconsistent translation results in low-resource language contexts. Certain Generative Pre-trained Transformers (GPT) and Multilingual Language Models (MLM) have attempted to address these issues through an expansive machine translation effort. The current LLMs are rife with problems in natural language processing: Especially, back translation, hallucination, and poor semantic reason functionality. Above all, the lack of cultural nuances limits ethnic Machine Translation (MT), GPT prompts, responses, and summarization. In addition, NLP is only orthographic with respect to low-used ethnic languages. In order to fix these challenges, remove the representation gap for low-resource languages in applications of AI, pave the way for more inclusive and culturally sensitive applications. We propose the AFA oracle of knowledge; the principle of 16, the design of a unified alphabet, the integration of phonemes, morphemes, graphemes, a comprehensive million-word dictionary, and quantum resistance cryptography into the development of a low-resource language model. The objective is to bring more dimension to the NLP parsing process, creating the much needed foundation model for low-resource languages. We extend an invitation

for collaboration to harness the rich linguistic diversity of ethnic languages, aiming to contribute to the advancement of AI while ensuring robust security and privacy in the imminent post-quantum era.

Notations

Notation	Explanation	Context
$\forall x \in \Sigma^*$	"For every string x in the set of all strings Σ^*."	Σ^* represents the set of all possible strings over the alphabet Σ, including the empty string ϵ.
$\exists P(x) \in L$	"There exists a property $P(x)$ such that $P(x)$ is a string in the language L."	Indicates a relationship or mapping between a string x and the language L through a property or function.
$x \in \Sigma^*$	"A string x belongs to the set of all strings over the alphabet Σ."	x is constructed by concatenating zero or more symbols from Σ.
ϵ	The empty string.	Represents a string with no symbols; $\epsilon \in \Sigma^*$.
$L \subseteq \Sigma^*$	"Language L is a subset of all strings over the alphabet Σ."	L is a specific set of valid strings defined by rules or grammar.
$f : \Sigma^* \to \Sigma^*$	"A function f maps strings in Σ^* to other strings in Σ^*."	Used for transformations, such as mapping between ethnic languages and English.
Σ_U	The unified alphabet.	Combines graphemes, phonemes, and morphemes of ethnic languages.
Σ_E	The English alphabet.	A finite set of 26 symbols plus extensions for modern usage.
$x[i : j]$	Substring of x from index i to $j - 1$.	Useful for parsing strings during transformation.
$P(x)$	A property or predicate applied to x.	Determines whether x satisfies certain conditions.
NLP (Natural Language Processing)	The field focused on enabling machines to understand, interpret, and generate human language.	Examples include machine translation, sentiment analysis, and speech recognition.
MLM (Masked Language Model)	A type of language model where certain words in a sentence are masked, and the model predicts the missing words.	Used in pretraining models like BERT to understand the context of language better.
Multilingual Language Model (MLM)	A model trained on multiple languages to perform cross-lingual understanding and generation tasks.	Examples include mBERT, XLM, and XLM-R, which support tasks like translation, sentiment analysis, and Q&A globally.
LLM (Large Language Model)	Large neural networks trained on vast amounts of text data to perform diverse language tasks.	Examples include GPT-4, BERT, and similar advanced models.
LrLM (Low-Resource Language Model)	A model designed to work with languages that have limited data or resources for training and evaluation.	Useful for preserving cultural and linguistic diversity, such as supporting underserved or ethnic languages.
$P(L)$	The power set of language L.	Contains all subsets of L, including $\emptyset$ and L itself.
$L_1 \cup L_2$	The union of two languages L_1 and L_2.	All strings that belong to either L_1 or L_2.
$L_1 \cap L_2$	The intersection of two languages L_1 and L_2.	All strings that belong to both L_1 and L_2.

L^R	The reverse of language L.	All strings in L with their symbols in reverse order.
L^n	The n-fold concatenation of L with itself.	If $n = 2$, then $L^n = LL$

2 Foundation, AFA the Principle of 16

This paper will showcase the knowledge gained through research of ancient computing systems and lexicon such as AFA, which is an oracle (black-box machine) of knowledge likened to a Turing machine, capable of solving both decision and function problems. We would like to show that ethnic languages offer a path to a unified alphabet which expands words using a set of strings over a finite alphabet following language rules. Until now, no known language (formal and informal) or computing system has demonstrated the capability to carry on many other strings (aligned and unaligned) in a meaningful cooperative process except for AFA. The AFA process is very straightforward. The instrument is made up of four strings, with four nodes (bisected Ugiri seeds) on each string. All binary representations from 0 to 15 correspond to a word in a set of 16 words or the AFA lexicon matrix (M_{afa}) of 16 elements (4 by 4). Two elements could be represented as (M_{ij}, M_{kl}): A relevant pair element of AFA the principle of 16, historically called "Ikpukpara" [3, 4, 5].

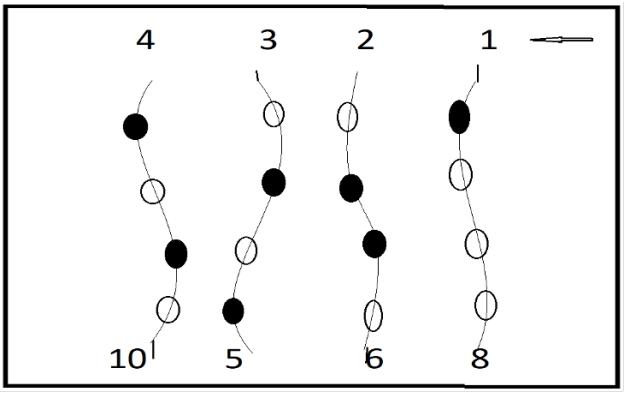

Figure 1: Sample cast-ed strings

In the event of a mishap, where something was stolen; a dream needs interpretation; mysterious signs or omen needs explanation; an individual needs to find their talent; Someone needs to know their life's outcome, and to make predictions in the future. The AFA oracle of knowledge is consulted. A would practitioner follows certain requirements to arrive at the right specification of the string ensemble. The nodes are made of special seeds from the Ugiri tree: This could be likened to the lithographic process of chip making. Normally, a practitioner would cast the dice to take the reading: They can go as deep as 3 iteration sets of enumeration. The cast above could be used as the first set. This was used at the bloom of human development in the South East, South West, and South region of the current day Nigeria. This computing system uses the principle of 16 to solve decision-and-function-related problems. Let us take a look at the intricate aspects of these problems.

2.1 Decision Problems

A decision problem is a problem with a yes/no answer. It can be formally represented as follows :

Definition 1 *A decision problem is a subset of a particular formal language $L \subseteq \Sigma_U^*$, where Σ_U^* denotes the set of all strings in some finite unified alphabet Σ_U. Given an input $x \in \Sigma_U^*$, the objective is to determine whether $x \in L$.*

Applying the same problem to the project, we have:

- **Problem:** Is a given string $x \in \Sigma_U^*$?
- **Formalization:** Define $L = \{x \in \Sigma_U^* : x \text{ is Ethnic}\}$.
- The problem is to decide if $x \in L$ when we define $L = \{x \in \Sigma_U^* : x \text{ is English}\}$.

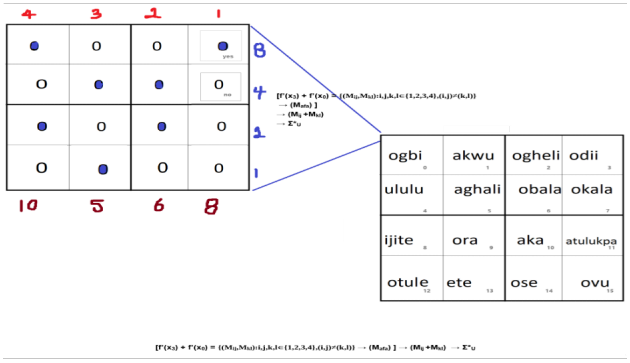

Figure 2: AFA matrix truth table

We argue that LLM ($L = \{x \in \Sigma_L^*\}$) is challenged by the identified problem: We want to prove that a unified alphabet (Σ_U) is more appropriate for inclusion, diversity, and equity. The result from the decision made in one step is a fractal of the final output. The battery of decisions or yes/no outcomes becomes an input in the transformational function stage: where the ovals true/false $(1/0)$ or open/close (yes/no), outcome stem from each string is represented in binary; thereafter mapped to the AFA matrix truth table with values in 16 language strings or words. This culminates in a decision and the synthesis of the decision's relationships to other events in the whole process.

2.2 Function Problems

A function problem requires computing a specific output rather than just a yes/no answer.

- A function problem is represented by a function $f : \Sigma_U^* \to \Sigma_U^*$, where the goal is to compute $f(x)$ for a given $x \in \Sigma_U^*$.
- Unlike decision problems, the output $f(x)$ is not limited to a Boolean value (true/false or yes/no).

The decision process follows a superset function f of functions f'. Let us represent elements of the set $S = \{w_1, w_2, w_3, \ldots, w_{16}\}$, where $w_n = 16$. These are the words found in the AFA lexicon matrix (Maf). The coordinated occurrences (f') map to these words or strings in the row-column major.

$$M_{afa} = \begin{bmatrix} M_{11} & M_{12} & M_{13} & M_{14} \\ M_{21} & M_{22} & M_{23} & M_{24} \\ M_{31} & M_{32} & M_{33} & M_{34} \\ M_{41} & M_{42} & M_{43} & M_{44} \end{bmatrix}$$

could be correlated within $f(x_{n=4}) = f'(x_0) + f(x_1) + f'(x_2) + f'(x_3)$ and enumerates as follows:

$$\begin{aligned}
f'(x_3) + f'(x_0) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^* \\
f'(x_0) + f'(x_2) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^* \\
f'(x_1) + f'(x_3) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^* \\
f'(x_1) + f'(x_0) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^* \\
f'(x_1) + f'(x_2) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^* \\
f'(x_3) + f'(x_2) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^* \\
f'(x_0) + f'(x_3) &= \{(M_{ij}, M_{kl}) : i,j,k,l \in \{1,2,3,4\}, (i,j) \neq (k,l)\} \to (M_{afa}) \to (M_{ij} + M_{kl}) \to \Sigma_U^*
\end{aligned}$$

If the formation below is the case:

$$
\begin{aligned}
f'(x_3) + f'(x_0) &= (\Sigma_L) \to \Sigma_L^* \\
f'(x_0) + f'(x_2) &= (\Sigma_L) \to \Sigma_L^* \\
f'(x_1) + f'(x_3) &= (\Sigma_L) \to \Sigma_L^* \\
f'(x_1) + f'(x_0) &= (\Sigma_L) \to \Sigma_L^* \\
f'(x_1) + f'(x_2) &= (\Sigma_L) \to \Sigma_L^* \\
f'(x_3) + f'(x_2) &= (\Sigma_L) \to \Sigma_L^* \\
f'(x_0) + f'(x_3) &= (\Sigma_L) \to \Sigma_L^*
\end{aligned}
$$

where $x \in \Sigma_L^*$. We raise a very important question at this juncture : Where does the ethnic alphabet Σ_U fit in [6].

2.3 The Sample Cast Reading

It is necessary to show a practical explanation of the reading from AFA in order to pique the mind and suggest other avenues for possible transformation and generation of exhaustive strings which would contribute to the current transformer architecture. Let us go through the logical translation of the sample cast shown earlier: We will concatenate (logical operator $+$) the elements (binary values) of the matrix open sets (dark ovals) arranged from right to left. Of course, the values are retained as two words $(M_{ij} + M_{kl})$. As mentioned earlier you will have to get the corresponding numbers of each string following binary 4 bit-wise. In the sample, we will enumerate thus:

$1000_2 + 1010_2$

$1 + 4 =$ ijite (8) aka (10) $\Leftrightarrow$ iweli aka ikenga elu : To be progressive. Σ_U^* expansion.

$1010_2 + 0110_2$

$4 + 2 =$ aka (10) Obala (6) $\Leftrightarrow$ ndi chizulu echizu : Titled men.

$0101_2 + 1000_2$

$3 + 1 =$ aghali (5) ijite (8) $\Leftrightarrow$ irado ive dika alo : Showing strong backing for something.

$0101_2 + 1010_2$

$3 + 4 =$ aghali (5) aka (10) $\Leftrightarrow$ ive na agba dika akpi : Something that stings like a scorpion or sand ant.

$0101_2 + 0110_2$

$3 + 2 =$ aghali (5) obala (6) $\Leftrightarrow$ Ndi echichi : Titled people.

$1000_2 + 0110_2$

$1 + 2 =$ ijite (8) obala (6) $\Leftrightarrow$ ụkwụ isi mmadu ike nala; To be alright.

$1010_2 + 1000_2$

$4 + 1 =$ aka (10) ijite (8) $\Leftrightarrow$ ana : land.

Note the first and the last stanza, these are the segue to the deeper meaning of the cast as the middle stanza decides the direction. The meaning does not always appear from the first cast: However, you can clearly see the direction when you carefully examine the loop. Here is the meaning of the cast-ed strings with Ugiri seeds as nodes: "One will become progressive in the land: Wherein, the support of powerful men will suffice. However, the so-called powerful ones will also become a stinging pain or obstacle."

The Afa principle of sixteen ensures that permutations of symbols in $\Sigma_U \to \Sigma_U^*$ can generate all possible strings:

P: $\Sigma_U \to \Sigma_U^*$

The Afa lexicon forms a complete basis for string generation: $\forall x \in \Sigma_U^*, \exists P(x) \in L$ This guarantees that any valid string $x \in \Sigma_U^*$ can be mapped to $L = \{x \in \Sigma_U^* : x \text{ is Ethnic }\}$.

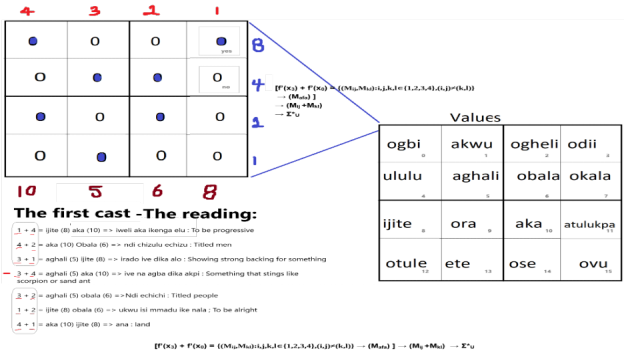

Figure 3: Cast-ed reading enumeration

2.4 Designing the Unified Alphabet

Unified alphabet ($\Sigma_U =$ A, B, CH, C, D, Ɖ, E, Ẹ, F, G, GB, GH, GW, H, I, Ị, J, K, KP, KW, Ḳ, L, M, N, Ṇ, NW, NY, O, Ọ, P, R, S, SH, we proceed to prove that all strings ($x \in \Sigma_U^*$) can be generated and transformed, across formal languages (L) and (L'), such as ethnic languages and modern English.

Mathematical Representation of the Unified Alphabet

2.4.1 Set of all strings over Σ_U

- $\Sigma_U^* = \{\epsilon,$ A, B, CH, ..., AAB, ABCH, ..., (all possible finite strings)$\}$
- This set encompasses phonemes, morphemes, and graphemes, forming a new orthography that represents ethnic languages and can absorb modern English structures.
- Subset $L \subseteq \Sigma_U^*$: L represents valid strings in a specific language under rules defined by cultural, phonological, and semantic constraints.
- Question to Prove: Given $x \in \Sigma_U^*$, does $x \in L$?

2.4.2 Step 1: Generative Capability of Σ_U^*

- The unified alphabet Σ_U is designed to incorporate all the linguistic elements (phonemes, graphemes, morphemes) necessary for both ethnic languages and English. This implies:

$$\forall x \in \Sigma^* E(\text{English strings}),\ x \in \Sigma_U \text{ because } \Sigma E \subseteq \Sigma_U.$$

- Additionally, since Σ_U contains all phonological and morphological constructs from ethnic languages:

$$\forall y \in \Sigma^* L(\text{Ethnic language strings}),\ y \in \Sigma_U^*$$

- Thus, Σ_U^* forms a superset of strings from both English and ethnic languages:

$$\Sigma^* E \cup \Sigma^* L \subseteq \Sigma_U^*$$

2.4.3 Step 2: Transformation Between Languages

- Define a transformation function $f : \Sigma_U^* \to \Sigma_U^*$ such that:

$$f(x) \in L,\ \forall x \in \Sigma_U^*$$

- The transformation f acts as a rule-based or probabilistic model (e.g., a neural algorithm or grammatical parser) that converts strings between languages.

- Example:
  - Input: $x =$ "KPALA" (This is Malayan)
  - Output: $f(x) =$ "HEAD" $\in \Sigma^* E$
- Since both inputs and outputs are drawn from Σ_U^* this function is well defined.

2.4.4 Step 3: Afa Permutations and Exhaustive Coverage

- The Afa principle of sixteen ensures that permutations of symbols in Σ_U can generate all possible strings:

$$P : \Sigma_U \to \Sigma_U^*$$

- The Afa lexicon forms a complete basis for string generation:

$$\forall x \in \Sigma_U^*, \exists P(x) \in L$$

- This guarantees that any valid string $x \in \Sigma_U^*$ can be mapped into L.

2.4.5 Step 4: Containment and Equivalence

From the above, we prove : **Containment:** For any string $x \in \Sigma^* E$ (English), since $\Sigma E \subseteq \Sigma_U$:

$$x \in \Sigma_U^* \Longrightarrow x \in L$$

Equivalence: The unified alphabet ensures mutual expressiveness:
$\Sigma_E^* \equiv \Sigma_L^*$ because the generative rules of L (ethnic languages) and L' (English) both draw from the same alphabet Σ_U.

By introducing the unified alphabet Σ_U, we demonstrated that:

1. Σ_U^* encompasses all possible strings from both English and ethnic languages.
2. The Afa lexicon, through permutations, generates all strings in Σ_U^*, ensuring comprehensive coverage.
3. Transformation f aligns strings across linguistic domains, preserving meaning and structure.

Hence, it is determined that the AFA lexicon and its permutations are generative in the most compounded form, it contains all possible strings for a given language with the unified alphabet (Σ_U). In our study, we will continue to experiment, explore, and decipher the unique patterns of sounds in the spoken words arising from strings over the alphabet of some ethnic languages. The unique phonological (phonemes), morphological (morphemes), and graphical (graphemes) structures of ethnic languages such as: Igbo, Yoruba, Hausa, and AAVE (Gullah/Geechee) languages cannot be overemphasized.

By focusing on the invariant nature of root words, vowel linking, tones, and their semantic enhancement through affixation, all of that fits well into a neural network of the AFA lexicon, Ogam lexicon, ethnic literature, religious texts, and number systems, which extends a million word dictionary. Classified textual data forms the syntax of the foundation model. All foundational data follow this process:

1. Classification (categorization)
2. Tokenization
3. Tagging
4. Stemming
5. Parsing
6. Semantic reasoning functionality

These are components of the early stage in the development of a low-resource language model. When engaging N-grams, activating the unigram, bigrams, trigrams, etc., in the proposed low-resource language model. There is a similarity to what we have always thought of with musical notes. The notes and chords form the harmonics: Do, ra, mi, fa, so, la, ti, do. This analogy to music gives rise to a new set of orthographies in machine human interface, which will enhance humanity, creating a foundation for GPT using cultural nuances in ethnic languages or voices.

2.5 Phonemes and morphemes rules in Igbo language

2.5.1 Morphology

This method allows us to break down the words into the smallest set of sub-words respecting the nature of the root words and the position of the vowels in the ethnic language. A possible sub-word is called a morpheme. In the Igbo language, the root words never change: they are constantly affirmed and qualified by another root word. Let us say, for example, the lungs → Ngụgụ in Igbo. We will see it as N-gụ-gụ. Why? When you apply that root: To comfort a crying child, you have to say: Gụ-gụ-ọ nwata ahụ. The most recent progression is the realization that there is a strong connection in all the meaningful words alluding to the fact that the root words never change but are defined by many other factors. Including whether the vowel is suffixed or prefixed. Words are only subject to semantic reason as another root follows them and as prefixed or suffixed in the order of vowel-consonant-vowel and vice-versa. This is typically an African word structure. Another example will be :

- (1) Remove it → Gụ-pụ ya
- (2) Bring it in → Gụ-ta ya.

From here it is a walk through the park because it gets easier. You get it now? If not, let this be your eureka moment: Gọ-zie → bless. In this case, the 'zie' Igbo trigram made the difference. Now, use the root word again with a different root (bigram): Gọ-rọ → remove as in oath. You can go further to understand that no matter what you do, the roots are the roots. This time, let us use the root and a trigram: Gọ-mụọ → to harness cosmic power. In conclusion, African languages are not optimized and fully represented in translations using LLM based on the subject-verb-object structure seen in English and other foreign tongues, which is characteristic of the creators of these LLMs in use today. As a result of this scientific exercise, we have developed an algorithm that captures the constraints of LLMs with supervised, unsupervised, mixed modes of machine learning, and reinforcement learning with human feedback (RLHF). Therefore, it is possible to perform many AI tasks in other ethnic languages, including Igbo, Yoruba, Hausa, and AAVE. The possibility of generating an infinite ensemble of strings that can represent any word you are speaking, will speak, and think of speaking is ever closer. Think of it as a giant matrix of word puzzles (already solved), if you will. We know that some strict instructions must be followed to achieve semantic reason functionalities. We want to share our knowledge as one approach to building a sustainable low-resource language model software. We are open to finding and working with those resources interested in using this to create the world of languages in their own image. There are other rigorous aspects to this.

2.5.2 Phonology

The method allows us to find sound patterns and structures in that spoken language. In relation to other natural or referenced sounds. These tone marks are used in most ethnic languages (Igbo and Yoruba) to contrast and compare tones, making it possible for sound patterns to be broken down into smaller units. However, phonemes are the smallest possible sound unit in which a language could be broken into: We see this in music as Do, Re, Mi, Fa, Sol, La, and Ti. Ethnic languages have non-exhaustive phonemes which are more than 7 musical notes. In an effort to properly represent any of these languages: It is imperative that we recognize the importance of getting the meaning from the set of strings in that particular language which is an element of the unified strings Σ_U^*.

Let us use the words 'one' or 'a' as an example in English: You can find an equivalent word and the meaning in an ethnic language following the unit of sound and its relations to cultural nuances. Tone marks are another aspect of language annotation. There are three (3) major tone marks that apply to these languages (Igbo and Yoruba) in question : Do= low-tone (\), Re= mid-tone/neutral (-), and mi= high-tone (/).

- a. Ò-fú → Òfú ≡ "One" or "A".
- b. Ò-kań → Òkań ≡ "One" or "A".

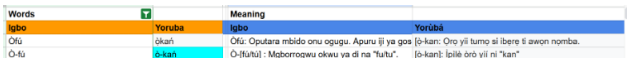

Figure 4: 1M word dictionary excerpt

Text and text-looking characters are the very foundation of a low-resource language model. These are also common with large language models. This means that we must first find the sources for our texts and classify them. We do not have many sources as these lowly-used languages lack resources on the internet. The majority of the dataset come from religious organizations including narrative that fits the religious literature designed for conversion. It becomes relevant

that we complete a dictionary of the chosen languages from the group that speak the languages. This is inclusive and culturally authentic. We designed an approach to accomplishing the tasks for this verbose dictionary; covering phonemes, morphemes, and graphemes.

3 Model Development with Shift-Left Approach

In the digital age, cybersecurity presents formidable challenges amid the exponential growth of attack surfaces and the emergence of artificial intelligence (AI). Recognizing the foundational role of anything language in the creation of the digital realm, we acknowledge the vulnerabilities inherent in existing security frameworks, which often yield a false sense of security.

Starting from the planning phase of the secure software development life cycle. This paper advocates for a paradigm shift in cryptography, moving towards a keyless approach that transcends conventional symmetric and asymmetric key primitives. Using some of the advanced cryptographic techniques, including post-quantum cryptography, we aim to embed authentication mechanisms and fortify data protection measures against evolving threats.

Central to our proposal is the integration of a low resource language model (LrLM) in NLTK, spaCy, Gensim, TextBlob, PyNLPI, CoreNL, etc. The library would be capable of keyless cryptography with natural language processing (NLP) and low resource machine learning (LrML) or deep learning methodologies. Through on-demand encryption and tokenization, we seek to redefine data security and privacy paradigms, particularly in low-resource languages. Our novel approach, embodied in the concept of a proper low-resource language model (LrLM), harnesses the tokenization capabilities of language models to safeguard sensitive information from malicious actors.

We emphasize the transformative potential of AI to augment computational capabilities for data protection. Using the vast computational power of AI, our goal is to take advantage of the data itself as a defense mechanism, thereby mitigates risks and ensuring robust cybersecurity measures.

3.1 Natural Language Processing

Natural Language Processing (NLP) refers to the branch of artificial intelligence (AI) that focuses on enabling computers to understand, interpret, and generate text (language) in a way that is both meaningful and contextually relevant. NLP encompasses a wide range of tasks, including text classification, sentiment analysis, machine translation, named entity recognition, and more.

All languages do not share a rich offline (use in the real world) and online (use in the virtual world) resource. As a result they have a low corpus footprint. When it comes to Low-resource Language Models (LrLM), NLP takes on a specific focus and set of challenges:

Low-resource Languages: Low-resource languages refer to languages with limited amounts of linguistic resources, such as annotated corpora, lexicons, and linguistic tools. These languages often lack the extensive data and resources available for major languages, making it challenging to develop NLP systems that perform well in such settings.

Language Modeling: In the context of LrLM, language modeling involves the building a statistical or neural network-based models that learn the underlying structure and patterns of a language from limited set of data. Language models for low-resource languages aim to capture the morphological, syntactic, and semantic characteristics of the language despite the scarcity of annotated data.

Resource Constraints: LrLM faces significant resource constraints, including limited training data, computational resources, and linguistic expertise. NLP techniques must be adapted to accommodate these constraints, focusing on approaches that are data efficient, computationally lightweight, robust to noise gaps, and variability.

Domain Adaptation and Transfer Learning: Given the scarcity of labeled data for low-resource languages, NLP approaches often leverage domain adaptation and transfer of learning techniques. Currently, the low resource language models are pretrained from the foundation laid by LLM which addresses the high-resource languages from within its algorithm: These models are later fine-tuned to fit the needs of the low resource language via machine translations. What we perceive is a commodity approach in translation transferred down to ethnic languages. The limited data for low resource languages exacerbates the inefficiency. Therefore, it is incorrect to say that accurate knowledge transfer and adaptation to the target linguistic context are enabled.

Community-driven Development: Development of NLP tools and resources for low-resource languages often relies on community-driven initiatives, involving collaboration with native speakers, computational linguists, local linguists, nd local organizations. Crowdsourcing, and participatory approaches play a crucial role in data collection, annotation, and evaluation for LrLM projects.

3.2 Low Resource Language Model (LrLM) Process

Alphabet Vocabulary as the Main Text Components: In LrLM, where annotated corpora may be scarce or nonexistent, using the alphabet vocabulary in the context of AFA oracle of knowledge is essential. This involves understanding the characters or symbols that comprise the language's writing system. By focusing on the alphabet vocabulary, LrLM can build models that operate at the character level, allowing for more robust text processing and generation, even in the absence of extensive linguistic resources.

Separating Vowels and Root Words (Special Case): For languages with complex morphology or agglutinative features, separating vowels and root words can be crucial for linguistic analysis and modeling. By identifying and isolating root words and vowels, LrLM can better understand the morphological structure of the language and generate accurate word forms and constructions.

Sub-words Formation: Sub word formation involves breaking down words into smaller linguistic units, such as morphemes or syllables. In LrLM, where vocabulary may be limited, sub-word modeling enables more effective utilization of training data and generalization to unseen word forms.

Tokenization: Tokenization refers to the process of segmenting text into individual tokens, such as words or sub-words (unigram, bigram, and trigram) for further analysis. In LrLM, tokenization is essential for processing text data and building language models. It enables efficient representation of linguistic units and facilitates downstream tasks such as language modeling and machine translation.

General Vocabulary and Syntax Formation (80 - 20%):

This parameter suggests a focus on general vocabulary and syntax formation, where approximately 80% of linguistic resources are dedicated to common vocabulary and syntactic structures, while the remaining 20% are allocated to less frequent or specialized terms. In LrLM, prioritizing general vocabulary and syntax formation ensures that language models capture the most common linguistic patterns and structures, improving their overall performance and usability in practical applications.

3.3 Data architecture

Decentralized Data Collection: Establish decentralized data collection nodes across the regions where Congo-Kwa languages are spoken. Engage local communities, computational linguists, and language experts to contribute linguistic data and samples. Utilize mobile applications or web platforms with offline capabilities to enable data collection in areas with limited connectivity.

Data Preprocessing and Cleaning: Develop a preprocessing pipeline to clean and normalize collected data, addressing noise, errors, and inconsistencies. Implement language-specific preprocessing techniques tailored to the linguistic characteristics of Congo-Kwa languages, such as tone normalization and morphological analysis.

Feature Extraction and Representation: Extract relevant linguistic features from the preprocessed data, considering phonetic, lexical, and grammatical aspects. Utilize distributed representation techniques such as word embeddings or contextualized embeddings (e.g., BERT) to encode linguistic information in a low-dimensional vector space.

Decentralized Model Training: Distribute model training across decentralized computing nodes to leverage distributed computing resources while minimizing centralized infrastructure. Train language models and prediction algorithms using federated learning or ensemble learning techniques to aggregate knowledge from diverse data sources while preserving data privacy and decentralization.

Language Augmentation and Prediction: Develop language augmentation algorithms capable of affixing, creating new, and predicting words in Congo-Kwa languages based on learned linguistic patterns. Incorporate probabilistic models and rule-based approaches to handle language morphology, syntax, and semantics for accurate word generation and prediction.

Resource Optimization: Implement resource-efficient algorithms and data structures to minimize computational and storage requirements, particularly in low-resource settings. Explore techniques for model compression, quantization, and knowledge transfer to reduce the footprint of deployed language model without compromising performance.

Continuous Evaluation and Improvement: Establish feedback loops for continuous evaluation of language augmentation and prediction models, soliciting input from native speakers and language experts. Incorporate mechanisms for model adaptation and retraining based on real-time feedback and evolving linguistic data.

By following this data architecture. It is possible to build a decentralized and resource-efficient system; to collect; to process; and use linguistic data input from low resource languages. This will work well for languages found

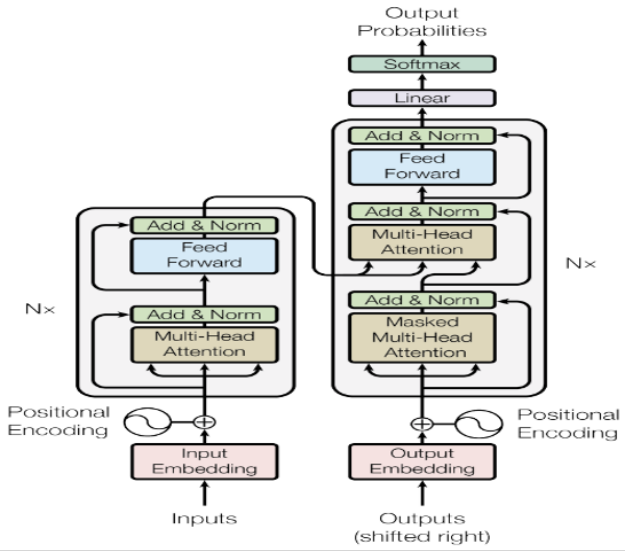

Figure 5: Attention is all you need [7]

in Congo-Kwa families, enabling language enhancement and prediction while respecting privacy and minimizing infrastructure requirements.

4 Transformers (Generative and Pre-trained)

Generative, and Pretrained Transformer for Congo-Kwa languages is a specialized language model trained to understand and generate text in the diverse languages of the Congo Basin region. Through pretraining and fine-tuning of Congo-Kwa language data, the model becomes proficient in generating coherent and contextually relevant text, contributing to language processing applications and linguistic research in the region.

Key components and definitions:

- **Generative Pre-trained Transformer (GPT)**: GPT refers to a type of deep learning model architecture based on the Transformer architecture, initially introduced by OpenAI. GPT models are capable of generating coherent and contextually relevant text based on input prompts or prompts given to them. These models are trained on large text corpora using unsupervised learning techniques, where they learn to predict the next word in a sequence of text given the preceding context. GPT models have been pre-trained on vast amounts of text data from diverse sources, enabling them to capture rich linguistic patterns and structures.
- **Congo-Kwa Languages**: Congo-Kwa languages are a group of languages primarily spoken in the Congo Basin region of Central Africa, including countries like the Democratic Republic of the Congo, Cameroon, and Gabon. These languages belong to the larger Niger- Congo language family and are characterized by their linguistic diversity, tonal features, and morphological complexity. - They make up the data needed for the NLP.
- **Language Understanding and Generation:** A GPT model for Congo-Kwa languages is trained to understand and generate text in these languages. It learns the syntactic, semantic, and contextual nuances specific to Congo-Kwa languages during the pre-training and fine-tuning stages. The model is capable of generating coherent and contextually appropriate text in Congo-Kwa languages based on input prompts or cues provided to it. It can perform a wide range of language processing tasks, including text completion, summarization, translation, and dialogue generation, tailored to the linguistic characteristics of Congo-Kwa languages.
- **Pretraining and Fine-tuning:** Pretraining involves training the GPT model on a large corpus of text data from Congo-Kwa languages, allowing it to learn the underlying linguistic patterns and structures. Fine-tuning further adapts the pre-trained GPT model to the specific linguistic features of Congo-Kwa languages, ensuring optimal performance for language understanding and generation tasks. We have not re-engineered anything at this time, except for what is needed on the text layer. The transformer architecture from Google: “Attention is all you need” [7]. This is still valid and aligns with the new approach for LrLM with tweaks.
- **Data Collection and Preprocessing:** Gather text data in Congo-Kwa languages from centralized and decentralized sources, including local communities, linguistic experts, and online repositories. Preprocess the

collected data to clean and normalize it, addressing noise, errors, and inconsistencies. Apply language-specific preprocessing techniques, considering the linguistic characteristics of Congo-Kwa languages such as phonological (tone) normalization and morphological analysis [8]. The taxonomy will definitely contribute to better outcomes for machine translation output for may AI application of NLP, MT, MLM, LrLM and etc, [9].

- **Data Representation and Feature Extraction:** Encode the preprocessed text data into numerical representations suitable for training a GPT model. Utilize techniques such as tokenization to convert words or characters into numerical tokens. Extract linguistic features from the data, capturing phonetic, lexical, and grammatical aspects relevant to Congo-Kwa languages [10].
  - **Model Architecture Selection:** Choose a GPT model architecture suitable for the task of language modeling in low-resource settings. Consider pre-trained GPT variants such as GPT-2 or GPT-3 as starting points, which can be fine-tuned on Congo-Kwa language data.
  - **Fine-tuning and Training:** Initialize the chosen GPT model with pre-trained weights and parameters. Fine-tune the GPT model on the preprocessed Congo-Kwa language data using transfer learning techniques. Train the GPT model using a language modeling objective, such as predicting the next word in a sequence given the preceding context. Employ continuous training strategies to update the GPT model with new data samples over time, ensuring adaptation to evolving linguistic patterns.
  - **Evaluation and Validation:** Evaluate the performance of the trained GPT model on held-out validation data, measuring metrics such as perplexity or accuracy. Validate the GPT model's linguistic quality and coherence through qualitative assessment by native speakers and linguistic experts.
  - **Deployment and Integration:** Deploy the trained GPT model as a language generation service or API, accessible for language augmentation and prediction tasks. Integrate the GPT model into the broader LrLM data architecture, enabling seamless interaction with decentralized data sources and continuous training mechanisms.
  - **Monitoring and Maintenance:** Monitor the performance and behavior of the deployed GPT model in production, tracking metrics and handling potential issues or drifts. Maintain the GPT model by periodically retraining or fine-tuning it with updated data, ensuring its relevance and effectiveness over time.
  - **Language Modeling:** GPT models are proficient at language modeling, which involves predicting the next word in a sequence of text given the preceding context. Within the data architecture, GPT can be employed to build language models tailored to Congo-Kwa languages, capturing their unique linguistic patterns and structures.
  - **Data Augmentation:** GPT models can generate synthetic text data that closely resembles natural language, allowing for data augmentation to increase the diversity and quantity of training data. In the context of low-resource languages, where labeled data may be scarce, GPT-generated text can supplement the training corpus, improving the robustness and performance of downstream NLP tasks.
  - **Word Prediction and Generation:** GPT models excel at predicting and generating coherent sequences of words based on input text. By fine-tuning GPT on Congo-Kwa language data, the model can predict new words, affixes, or phrases within the linguistic context, facilitating language expansion and enrichment.
  - **Contextual Embeddings:** GPT produces contextualized word embeddings that capture the meaning of words in relation to their surrounding context. These embeddings can be utilized within the data architecture to encode linguistic features and representations, enhancing the accuracy and richness of language processing tasks.
  - **Adaptation to Low-Resource Settings:** GPT models can be adapted and fine-tuned on limited amounts of labeled data, making them suitable for low-resource language scenarios.Fine- tuning GPT on Congo-Kwa language data allows the model to capture the specific nuances and characteristics of these languages, improving performance in tasks such as language modeling and generation.

5 Continuous training

Initial Model Training: Initially, the architecture involves training a language model (LM) on available linguistic data from Congo-Kwa languages. This training process involves feeding the LM with sequences of words from the training corpus and adjusting the model's parameters (e.g., neural network weights) to minimize the prediction error. The commodity approach attached to ethnic or indigenous knowledge could case disenfranchising of these local authorities which would perpetuate natural language processing and endangerment to the communities of lowly used languages [11]. We will engage these communities as we go through the stages of development.

Fine-tuning with Continuous Training: After the initial model training, continuous training becomes essential for adapting the LM to evolving linguistic patterns and expanding linguistic knowledge. Continuous training involves periodically updating the LM using new data samples collected from decentralized sources or user interactions. For LrLM, continuous training is particularly crucial due to the scarcity of labeled data. By continuously incorporating new data, the LM can adapt to the linguistic characteristics of Congo-Kwa languages more effectively.

Incremental Learning: Continuous training facilitates incremental learning, allowing the LM to incrementally update its parameters based on newly acquired knowledge. This incremental learning process helps the LM remain up-to-date with the latest linguistic trends and variations in Congo-Kwa languages, ensuring its relevance and accuracy over time.

Adaptation to User Preferences: Continuous training also enables the LM to adapt to user preferences and feedback. By incorporating feedback from users or linguistic experts, the LM can refine its predictions and language generation capabilities to better align with user expectations and linguistic norms.

Dynamic Model Updating: In a decentralized architecture, where data sources may vary in quality and quantity, continuous training allows for a dynamic model updating based on the availability and relevance of new data. The LM can prioritize training on data from reliable sources or regions with significant linguistic diversity, ensuring balanced and representative model updates.

Performance Monitoring and Evaluation: Continuous training involves monitoring the LM's performance metrics, such as perplexity or accuracy, over time. Regular evaluation of the LM's performance helps identify areas for improvement and guides the selection of training strategies and data sources for optimal model refinement. In summary, training, including continuous training, plays a pivotal role in the LrLM data architecture by enabling the adaptation, refinement, and expansion of language models for low-resource languages like those found in Congo-Kwa language families. Continuous training ensures that the LM remains relevant, accurate, and capable of addressing evolving linguistic challenges and user needs in a decentralized and resource-constrained environment.

6 Keyless encryption in large language models

Keyless encryption implies end-to–end, on-demand encryption for timeless security. This simply means that it is resistant to quantum assailants' attacks and very easy to apply without reshaping the infrastructure. More so, it ascertains that all entities that require authentication no longer have to move, reuse, store, and manage keys. This offers a versatile and secure approach to protecting linguistic data in LLM applications, addressing concerns related to data security, privacy, and confidentiality in resource-constrained linguistic environments. It is possible to perform computation over encrypted data with an end-to-end cryptographic system for cloud, IoT, and mobile devices (ECSMID) and DatashieldAI.

- **Data Storage and Transmission:** LLMs often operate in decentralized or distributed environments, where data storage and transmission may occur across multiple nodes or devices. Keyless encryption can be employed to encrypt linguistic data stored on local devices, servers, or cloud repositories, protecting it from unauthorized access or interception during transmission.
- **Privacy-Preserving Collaboration:** In collaborative research or data sharing initiatives involving multiple stakeholders, keyless encryption enables privacy-preserving collaboration. Linguistic data shared among researchers, linguists, and language experts can be encrypted without the need for shared keys, ensuring data confidentiality while facilitating collaboration.
- **User Privacy in Language Processing Applications:** Language processing applications powered by LLMs often involve user interactions, such as text input or voice commands. Keyless encryption can be applied to protect user-generated linguistic data, preserving user privacy and confidentiality, especially in applications involving sensitive or personal information.
- **Secure Model Deployment and Inference:** Deployed LLMs may process sensitive linguistic data in real-time, raising concerns about data security and privacy. Keyless encryption mechanisms can be integrated into LLM deployment pipelines to encrypt input data and model outputs, ensuring end-to-end data protection without the need for cryptographic keys.
- **Data Anonymization and Pseudonymization:** In scenarios where linguistic data needs to be anonymized or pseudonymized to comply with privacy regulations or ethical considerations, keyless encryption provides a secure means of anonymization. Linguistic identifiers and personally identifiable information (PII) can be encrypted without relying on cryptographic keys, preserving data anonymity while maintaining usability.
- **Cross-Language Data Fusion:** LLMs may aggregate linguistic data from multiple languages or dialects for cross- language modeling or translation tasks. Keyless encryption can be applied to ensure the security

and privacy of multi-lingual datasets, allowing for secure data fusion and integration without the need for language-specific encryption keys.

7 Large Language encryption Model (LLeM) - A special case

Large Language Encryption Model (LLeM) is a sophisticated language model that integrates encryption capabilities into its architecture, enabling the secure processing, generation, and transmission of textual data while preserving data confidentiality and privacy [12].

7.1 Encryption can be applied in both Large Language Models (LLMs) and Low-resource Language Models (LrLMs)

Large Language Models (LLMs): Data Security in Decentralized Environments: LLeM can be used in LLMs operating in decentralized or distributed environments to encrypt linguistic data stored on local devices or transmitted between nodes. This ensures data security and confidentiality, especially in collaborative research or data-sharing initiatives. User Privacy in Language Processing Applications: LLeM can protect user-generated linguistic data in language processing applications, preserving user privacy and confidentiality during text input or interaction with LLM-powered systems.

Low Resource Language Models (LrLMs): Secure Data Transmission and Storage: In LrLMs where linguistic data may be scarce and valuable, LLeM can encrypt data during transmission and storage, safeguarding it from unauthorized access or interception. These enhance data security and confidentiality in resource-constrained linguistic environments. Privacy-Preserving Collaboration: LLeM enables privacy-preserving collaboration among stakeholders involved in LrLM development or research. By encrypting linguistic data shared among researchers, linguists, and language experts, LLeM ensures data confidentiality while facilitating collaborative efforts.

8 Natural language toolkit (NLTK)

NLTK, short for Natural Language Toolkit, is an open-source Python library designed to facilitate the exploration, processing, and analysis of human language data. It provides a comprehensive suite of tools, resources, and algorithms for various natural language processing (NLP) tasks, making it a valuable asset for researchers, educators, and practitioners in the field of computational linguistics and artificial intelligence.

NLTK serves as a versatile and accessible platform for conducting linguistic research and developing NLP applications across diverse domains. Its extensive collection of modules and functionalities enables users to perform a wide range of tasks, including text preprocessing, tokenization, part-of-speech tagging, syntactic parsing, semantic analysis, and machine learning-based classification.

At the core of NLTK's capabilities lies its modular architecture, which allows users to seamlessly combine and customize NLP components to suit their specific research objectives or application requirements. Whether analyzing large text corpora, building language models, or developing language processing pipelines, NLTK offers a flexible and user-friendly environment for experimentation and innovation.

Furthermore, NLTK's emphasis on educational resources and documentation makes it an invaluable tool for learning about NLP concepts and techniques. Through tutorials, examples, and interactive demos, NLTK empowers both novice and experienced users to gain insights into the inner workings of natural language processing systems and explore advanced topics in computational linguistics.

NLTK stands as a cornerstone of the NLP community, providing researchers and practitioners with the tools and resources needed to tackle complex linguistic challenges, advance the state-of-the-art in language technology, and unlock new opportunities for understanding and interacting with human language data.

8.1 NLTK Functionalities for LrLM

When the process is completed, it will become the trigger for the library that most low-resource languages could possibly use.

1. Classification: Classification involves categorizing text documents into predefined classes or categories based on their content. In NLTK, classification tasks often involve training machine learning models, such as Naive Bayes classifiers or Maximum Entropy classifiers, on labeled text data to predict the category of unseen documents.

2. Tokenization: Tokenization is the process of splitting text into individual tokens, such as words, phrases, or sentences, for further analysis.NLTK provides tokenization functions to segment raw text into tokens based on specific criteria, such as white space, punctuation, or grammatical rules.
3. Stemming: Stemming involves reducing words to their root or base form by removing affixes or suffixes. NLTK includes various stemming algorithms, such as the Porter Stemmer or Snowball Stemmer, which can normalize words to their root forms to improve text analysis and information retrieval tasks.
4. Tagging: Tagging refers to the process of assigning part-of-speech tags or labels to words in a text corpus. NLTK offers part-of-speech tagging functionalities, allowing users to assign tags such as noun, verb, adjective, or adverb to words based on their grammatical roles within sentences.
5. Parsing: Parsing involves analyzing the grammatical structure of sentences to identify syntactic relationships between words. NLTK includes parsers for tasks such as constituency parsing and dependency parsing, enabling users to analyze sentence structures and extract syntactic information from text data.
6. Semantic Reasoning Functionalities: Semantic reasoning involves understanding the meaning and context of text beyond its surface-level representation. NLTK provides tools and resources for semantic analysis tasks, such as igboapi.com, which offers lexical databases and semantic similarity measures for exploring word meanings and relationships.

9 Implementation steps of the proposed algorithm

9.1 Representations of different embeddings

Upper case	A	B	CH	C	D	Đ	E	Ẹ	F	G	GB	GH	GW
	H	I	Ị	J	K	KP	KW	ħ	L	M	N	Ṇ	NW
	NY	O	Ọ	P	R	S	SH	Ṣ	T	TS	U	Ụ	V
	W	Y	Ỵ	Z									

Lowercase	a	b	ch	c	d	ħ	e	ẹ	f	g	gb	gh	gw
	h	i	ị	j	k	kp	kw	ħ	l	m	n	ṇ	nw
	ny	o	ọ	p	r	s	sh	ṣ	t	ts	u	ụ	v
	w	y	ỵ	z									

Unigram a, b, d, e, f, g, m, n, o, ọ, p, r, s, t, y, z,

b. Bigrams	Ja,	na,	bi,	ch,	di,	fi,	gb,	gh,	gi,	gw,	ji,	ki,	kp,	kw,	mi,	ni,
	nw,	ny,	oo,	ri,	si,	so,	ti,	sh,	yi,	zi,						

—> jo
—> ju

The goal is to arrive at all possible bigrams. If possible, as a part of text-to-speech tag and vice-versa.

c. Trigrams	bia,	chi,	die,	fio,	gbu,	gha,	gii,	gwu,	jie,	kia,	kpa,	kwa,	mia,	nia,	nwa,	nye,
	ooo,	rie,	sie,	tie,	shi,	yie,	zie,									

The goal is to arrive at all possible trigrams. If possible, as a part of text-to-speech tag and vice-versa.

d. Vowels	A	E	I	Ị	O	Ọ	U	Ụ
	a	e	i	ị	o	ọ	u	ụ

9.2 Create short sentences (organic syntax synthesis by sub-words)

- I di na ya / nia
- I so ni-a/na ya
- Ani/ala/ali oma
- Nwa oma
- Obi ojo
- Ndi si njo
- Odi nma

Create longer sentences (organic syntax synthesis by concatenation)

- Nwa oma ani/ala oma
- Ndi si na njo di nma
- Obi ojo, Idi nia/naya
- Nwa oma obi ojo
- O-di nma na I so nia/na ya

Other data sets or corpus (larger collections):

- Ogam dictionary
- Regular dictionary
- Afa Lexicon
- Igboapi.com

Tokenization

You can break the large corpus down again to sub-words and then to numbers in order to facilitate ML and MT. We will be using the example below for clarity of the non-repetative nature of tokens.

Example sentence:

O-di nma na I so ni-a/na ya

Tokenized into sub-words:

['O-di', 'nma', 'na', 'I', 'so', 'ni-a', '/', 'na', 'ya']

Token to number mapping:

{ 'I': 0, 'so': 1, 'ya': 2, 'ni-a': 3, 'nma': 4, '/': 5, 'O-di': 6, 'na': 7 }

Converted numbers:

[6, 4, 7, 0, 1, 3, 5, 7, 2]

Stemming

The idea is to engage some mechanism better described as vowel suffix or prefix to root linking to reinforce word bases and meanings.

Examples: o-gb, a-gh, gi-a, gw-a,
a-bi, ch-e, a-di, e-fi, gb-o, gh-a, a-gi, a-gw,
ji, ki, kp, kw, mi, ni, nw, ny, oo, ri, si,
sh-o, a-zi,
aso, oti, o-sh, oyi, zi-a

Tagging

Make the word corresponding to a particular part of Igbo speech tagging.

Conjugation of verbs:

je to walk/go
ga to walk/go
gọ to harness

- Je → ije → eje → na-eje
- Ga → iga → aga → na a-ga → ọ-ga-la

- Gọ → igọ → agọ → na a-gọ → ọ-gọ-la

  ****
  Go → going → is going → gone
  Lọ → lati lọ → m lọ | n lọ → lọ
  ****

Looking at this, you can already notice that there are similarities in the Yoruba language.

Parsing

— It must be expanded to the English grammar part of speech (NAVAPCIP) structure.

1. Noun – Tom lives in New York.
2. Pronoun – Did she find the book she was looking for?
3. Verb – I reached home.
4. Adverb – The tea is too hot.
5. Adjective – The movie was amazing.
6. Preposition – The candle was kept under the table.
7. Conjunction – I was at home all day, but I am feeling very tired.
8. Interjection – Oh! I forgot to turn off the stove.

Which in other languages can be expanded in the specific speech structure. E.g

Oooo, e-che-kwa-m na i-je-re ma-na ha si na i-bu nwa o-ma o-bi o-jo na ina ejeka.
N we-lu si ke-du e-be i-na e-je-gi?

Semantic Reasoning Functionalities

This should be the last stage: You create inference rules to capture new facts (syntax). Human feedback is obtained through the community effort. New textual samples are used to train the machine: The idea is that, inferences are drawn from the existing and the new syntax samples fed to the machine to promote reasoning functionalities for machine and human understanding and nuanced clarity. - This is where reinforcement learning comes in.

> "Nwa oma obi ọjo kedu ebe ina e jegi?"

This is semantic artificial intelligence: Adding context, knowledge, and valuable insight into the data set supplied earlier. With reinforcement learning and human feedback, we can easily speed up low-resource languages into wider application of artificial intelligence for the most part.

10 Neural Network

As a native speaker of the Igbo language, the writer knows the similarities in sound units and sub-words with other neighboring languages. It has computational linguistic equivalence to Yoruba, and many other Congo-Kwa languages. We would build a giant matrix of words in our development phase. The preliminary is to help you understand how words formulate into sentences from Igbo, romanized alphabets. These are considered to be foundational vocabulary in most ethnic languages. The Igbo language $(L \subseteq \Sigma^* U)$ rules are followed.

Let us start with the native romanized alphabets:

> A, B, CH, C, D, Ɗ, E, Ẹ, F, G, GB, GH, GW, H, I, Ị, J, K, KP, KW, ɦ, L, M, N, Ṇ, NW, NY, O, Ọ, P,
> R, S, SH, Ṣ, T, TS, U, Ụ, V, W, Y, Ỵ, Z.

8 of these alphabets make up what is known as vowels: A, E, I, Ị, O, Ọ, U, Ụ.

The rest are 35 known as consonants and diacritics: B, CH, C, D, Ɗ, Ẹ, F, G, GB, GH, GW, H, J, K, KP, KW, ɦ, L, M, N, Ṇ, NW, NY, P, R, S, SH, Ṣ, T, TS, V, W, Y, Ỵ, Z.

There are in the last group 2 pseudo-vowels: M and N.

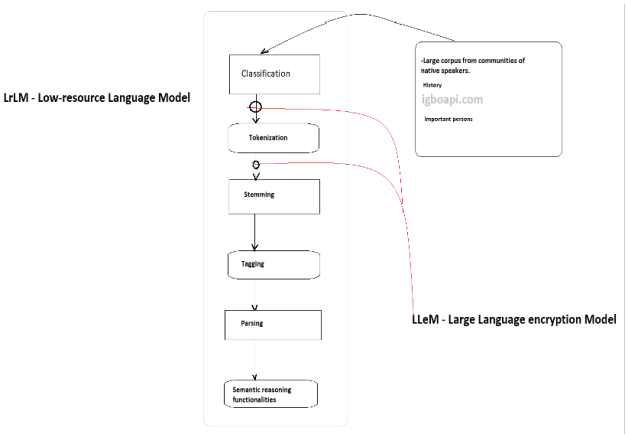

Figure 6: Implementation stages

With the above insight, we can now create an exhaustive list of values arranged in a matrix following the algorithm below: These are some of the consonants: B, CH, D, F, G, GB, GH, GW, H, J, K, KP, KW, L, M, N, Ṅ, NW, NY, P, R, S, SH, T, V, W, Y, Z.

1. Using vowels as prefixes, concatenate them with the consonants:
   - aB, aCH, aD, aF, aG, aGB, aGH, aGW, aH, aJ, aK, aKP, aKW, aL, aM, aN, aṄ, aNW, aNY, aP, aR, aS, aSH, aT, aV, aW, aY, aZ.

   Continue iteration with other vowels.
2. Using vowels as suffixes, concatenate them with consonants:
   - Ba, CHa, Da, Fa, Ga, GBa, GHa, GWa, Ha, Ja, Ka, KPa, KWa, La, Ma, Na, Ṅa, NWa, NYa, Pa, Ra, Sa, SHa, Ta, Va, Wa, Ya, Za.

   Continue iteration with other vowels.
3. Concatenate the vowels, against vowels. By default, vowels start most sentences in Igbo. There are cases when they are used as interjections and they could recur:
   - Aa, ee, ii, etc.
4. Identify all bigrams and trigrams there are:
   - a. Ba, da, fa, ga,
   - b. Cha, Gba, The, Gwa, ...
5. Identify the root words (bigram and trigram) as they do not change in meaning but vary based on the vowel's positioning. When the vowel is in the prefix (front) of a consonant, together they mean one thing and when the vowel is at the suffix they mean another.
   - For example: ga (meaning one thing), aga (meaning another).
6. Create vowel-consonant-vowel word formation with these root words in the last step. Show the possible combinations available with vowels as a prefix, assuming all combinations are considered.
   - E.g., Aba, acha, Ada, Afa, Aga, etc.
   - Include pseudo-vowels in this prefix from root iterations in Step (5).
7. With these generated, it would be relevant to combine or concatenate those words:
   - Aba acha, Aba ada, Aba Afa, etc.
8. Form short sentences by combining vowels alone and the words you created to give a short sentence.
   - E.g., o Aba acha, o Aba ada, o Aba Afa, etc.
   - A word must be activated by a vowel to create intelligent strings.
9. Combine those short sentences for longer sentences.
   - E.g., o Aba acha, o Aba ada, o Aba Afa dafa socha, etc.

10. We will then supply you with a language data set to see how well you have done.
    - This will include samples of larger corpora for validation.

This is one of the tracts on how to build a giant neural network from which the words will only be activated if they follow the right order of vowel-consonant-vowel, consonant-vowel-consonant, and the specific language rules. In iterations, the sample syntax will show the possible permutations and combinations [5]. Congo-Kwa languages exhibit this order of formation. Practically, word meanings are hidden in those spoken (sound) words.

11 Optimizing Tokenization for Low-Resource Languages: Clarity and innovation

Tokenization, the process of splitting text into smaller units (tokens), is crucial for natural language processing (NLP). For low-resource languages, tokenization faces unique challenges, including limited data, diverse morphological structures, and underrepresentation in mainstream language models. Here is how tokenization can be optimized for low-resource languages, with examples for clarity and innovative solutions.

11.1 Understand the Challenges of Low-Resource Languages

Low-resource languages often have characteristics that make tokenization complex:

- **Rich Morphology:** Words may have multiple affixes, requiring segmentation (e.g., Swahili: Ninapenda = Ni (I) + na (present tense) + penda (love)).
- **Compound Words:** Languages like German or Igbo may have long compound words that must be split into meaningful subcomponents.
- **Lack of Predefined Tokenizers:**Most tokenizers are trained on high-resource languages (e.g., English, French), which don't always generalize well to low-resource languages.
- **Scripts and Orthography:** Some languages have unique scripts or nonstandardized orthographies that complicate tokenization.

11.2 Tokenization Techniques for Low-Resource Languages

There are many techniques available in tokenization processes. However, we find that some techniques work best for a family of languages, certainly follows the language rules, and peculiar beyond any shallow approach devoid of a rich vocabulary.

11.3 Character-Level Tokenization

- **Explanation:** Breaks text into individual characters. This is simple and effective for languages with complex morphology.
- **Example:**
  - Text:Gbaraghari
  - Tokens:['G', 'b', 'a', 'r', 'a', 'g', 'h', 'a', 'r', 'i']
  - Tokens:['Gb', 'a', 'r', 'a', 'gh', 'a', 'r', 'i']
- **Advantages:**
  - Captures all possible words, including rare and unseen ones.
  - Reduces the risk of out-of-vocabulary (OOV) words.
- **Disadvantages:**
  - Produces longer sequences, which may increase computational costs.

11.4 Subword Tokenization

- **Explanation:** Splits text into smaller units like prefixes, suffixes, and roots using methods like Byte Pair Encoding (BPE) or SentencePiece.
- **Example:**
  - Text: Gbaraghari

  - Tokens: ['Gba', 'ra', 'gha', 'ri']
- **Advantages:**
  - Balances the trade-off between character-level and word-level tokenization.
  - Handles rare words better by breaking them into common subwords.
- **Disadvantage**
  - Requires a well-designed vocabulary to suit the language's structure.

11.5 Morphological Tokenization

- **Explanation:** Use morphological analysis to split words into their smallest meaningful units.
- **Example:**
  - Text: Gbaraghari
  - Tokens:['Gba' (shoot), 'ra' (past tense), 'gha' (scatter),'ri' (present tense)]
  - Tokens:['Gba' (shoot), 'ra' (existed), 'gha' (scatter),'ri' (rising)]
- **Advantages:**
  - Preserves semantic meaning and linguistic integrity.
  - Particularly useful for languages with rich morphology.
- **Disadvantage:**
  - Requires linguistic expertise and resources to build morphological analyzers.

11.6 Hybrid Tokenization

- **Explanation:** Combines multiple approaches, such as character-level tokenization for rare words and subword/morphological tokenization for common words.
- **Example:** Swahili.
  - Text: Ninapenda vijana wazuri.
  - Tokens: ['Ni', 'na', 'penda', 'vi', 'jana', 'wa', 'zuri']
- **Advantages:**
  - Maximizes flexibility and efficiency.
  - Tailored to the specific needs of the language.

11.7 Innovations in Tokenization for Low-Resources Languages

11.8 Leveraging Unified Alphabets and Scripts

- **Explanation:**Standardizing the script and orthography for related languages or dialects to improve tokenization consistency.
- **Example:**For African languages, using a unified orthography for Bantu languages can simplify tokenization across multiple languages.

11.9 Context-Aware Tokenization

- **Explanation:** Incorporate context into tokenization decisions using neural models that account for syntax and semantics.
- **Example:** Neural tokenizers can differentiate between homographs like bank (a riverbank vs. a financial institution) in context.

11.10 Data Augmentation

- **Explanation:** Generate synthetic data to expand the dataset, improving the training of tokenizers.
- **Example:** For Igbo, create variations of sentences by swapping synonyms or altering sentence structures:
  - Original: Nwaanyi ahụ na-eri nri. (The woman is eating food.)
  - Augmented: Onye ahụ na-eri nri. (The person is eating food.)

11.11 Incorporating Phonology and Morphology

- **Explanation:** Use phonological rules and morphological patterns to inform tokenization.
- **Example:** For tonal languages like Yoruba, include tone marks as features during tokenization:
  - Text: ọkọ (husband) vs. òkò (hoe).

11.12 Community-Driven Tokenizer Development

- **Explanation:**Engage native speakers and linguists in the tokenization process to ensure cultural and linguistic accuracy.
- **Example:**Crowdsourcing annotations and tokenization rules for underrepresented languages.

11.13 Machine Translation for Tokenization

- **Explanation:** Use translation models to align tokenization strategies across languages.
- **Example:** Translate a sentence in Hausa into English to identify tokenization rules, then apply those rules to Hausa.

11.14 Example Workflow for Tokenizer Development

- Step 1: Collect and Clean Data
  - Use texts from books, social media, or transcribed speech.
  - Remove noise like misspellings and non-standard characters.
- Step 2: Select a Tokenization Strategy
  - Choose character-level, subword, or hybrid tokenization based on language complexity and available resources.
- Step 3: Train the Tokenizer
  - Use tools like Hugging Face's Tokenizers, SentencePiece, or spaCy for tokenizer training.
- Step 4: Evaluate and Iterate
  - Evaluate the tokenizer using metrics like perplexity and out-of-vocabulary rate.
  - Iterate by adjusting the vocabulary size, subword splitting rules, or morphological patterns.

12 Conclusion

We presented a unified alphabet, with containment, equivalence, the capability to absorb preexisting, high-resource languages which at first did not recognize the ethnic voices and in-built security for prompts and responses. The inaccuracy of machine translation (MT) from say English to Igbo, Yoruba, Hausa and African American Vernacular English (AAVE) is directly linked to the challenges of back translation and core of large language model (LLM) misalignment with other ethnic languages. -This is the bane of NLP with respect to ethnic languages. The language of the current LLMs, though equivalent, contains finite alphabets of their creators. Those language strings do not include ethnic strings. Therefore, they are irrelevant for accurate transformation in MT. Using AFA matrix, which shares a lot in common with neural based LLM as shown. We presented the principle of 16 with the most interesting string expansion; statistical analysis and synthesis of the next-best outcome in a set of strings over a unified alphabet where decisions are mapped into functions and vice versa. This yields a non-exhaustive list of strings by way of permutation. This paper presents a comprehensive framework for advancing data security through the convergence of AI, cryptography, and language modeling. Using innovative technologies and redefining traditional paradigms, our goal is to empower organizations and individuals to be heard in their own languages while safeguarding their data in an increasingly interconnected and digitized world.

References

[1] Pratik Joshi, Sebastin Santy, Amar Budhiraja, Kalika Bali, and Monojit Choudhury. The state and fate of linguistic diversity and inclusion in the nlp world. *arXiv preprint arXiv:2004.09095*, 2020.

[2] Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. Natural language processing: state of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744, 2023.

[3] Alexander Alfred Onukwube Anaedo. Afa (divination): The mouthpiece of the unseen. Scribd, 2008. Retrieved from `https://www.scribd.com/document/412918926/t-Anedo-Alexander-Alfred-Onukwube`.

[4] Victor Manfredi. On the verb's edge in igbo and yoruba. Tech report, Boston University, 1997. [PDF].

[5] Austin J Shelton. The meaning and method of afa divination among the northern nsukka ibo. *American Anthropologist*, 67(6):1441–1455, 1965.

[6] S. Albarino. Can low-resource languages catch up in the multilingual model race? new report. Slator, July 6 2023. Retrieved from `https://slator.com/can-low-resource-languages-catch-up-multilingual-model-race/`.

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

[8] A Conneau. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*, 2019.

[9] Philipp Koehn and Rebecca Knowles. Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*, 2017.

[10] Jacob Devlin Ming-Wei Chang Kenton and Lee Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of naacL-HLT*, volume 1, page 2. Minneapolis, Minnesota, 2019.

[11] Steven Bird. Decolonising speech and language technology. In *28th International Conference on Computational Linguistics, COLING 2020*, pages 3504–3519. Association for Computational Linguistics (ACL), 2020.

[12] Lily Chen, Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray A Perlner, and Daniel Smith-Tone. *Report on post-quantum cryptography*, volume 12. US Department of Commerce, National Institute of Standards and Technology . . . , 2016.