Optimizing a Minimal Language using Pre-trained Language Models

Ajay Narayanan

Technical University of Munich

Agenda

Introduction

The Purpose of Language

- Human language evolved to aid survival and propagation.
- Facilitates collaboration, competition, and influence.
- Language is shaped by the trade-off between:
  - **Expressiveness**
  - **Efficiency** (driven by evolution)

Why Natural Languages Are Not Optimal

- Real-world communication happens in noisy environments.
- Redundancy helps with error correction.
- Natural languages are constrained by:
  - Incremental language processing
  - Need for compositionality, systematicity, and concatenation
- Not optimal in the information-theoretic sense.

Constructed Languages (ConLangs)

- Intentionally created languages.
- Types:
  - Auxiliary (e.g. Esperanto)
  - Fictional (e.g. Dothraki, Quenya, Klingon)
  - Logical (e.g. Lojban)
  - Minimalist (e.g. Toki Pona)
  - Expressive (e.g. Ithkuil)

What Does "Optimal" Mean for a Language?

- Information-theoretic efficiency
- Ease of learning
- Expressiveness
- Robustness in noisy conditions
- Ability to convey complex ideas
- **Key Questions:**
  - Can we define and measure these?
  - Can we build languages optimizing for one or more of these constraints?

Can LLMs Help Design Better Languages?

- LLMs encode deep patterns in natural language and world knowledge.
- They may uncover latent linguistic structures.
- Potential use in evaluating or designing:
  - Phonology, orthography, morphology
  - Syntax and vocabulary
- Could LLMs help define and optimize linguistic trade-offs?
- Could LLMs assist in the language generation process?

Thesis Goals

1. Define linguistic optimality across dimensions.
2. Explore design of an efficient constructed language.
3. Investigate how LLMs can aid this process.

Methodology

Methodological Overview

- Step-by-step approach inspired by language construction literature.
- Language specification includes:
  - Phonemic inventory
  - Phonotactics
  - Grammar
  - Vocabulary
- Translation of known texts used for evaluation.

Research Design

- Modular, computational experimental design.
- Objective: Guide LLMs to generate human-like ConLangs.
- Pipeline consists of:
    - Phonology
    - Morphology
    - Syntax
    - Vocabulary
- Allows ablative analysis and future extensibility.

Language Generation Pipeline

- Modules operate sequentially or in parallel with shared dependencies.
- Each module is an independent variable.
- Enables targeted experimentation to study:
  1. Effect of phoneme inventory size
  2. Influence of phonotactic rules
  3. Impact of grammatical structure
  4. Variation in vocabulary generation

Evaluation Framework

- Evaluation modules benchmark generated languages.
- Results stored alongside each language.
- Three main categories:
  - Information Loss
  - Simplicity
  - Zipf's Law Compliance

Information Loss Metrics

- **Machine Translation Scores**:
  - Round-trip translation using LLM
  - Evaluated with BLEU, ROUGE, METEOR
- **Reading Comprehension**:
  - LLM answers questions based on original and translated texts
  - Score = % of correct answers

Simplicity Metrics

- **BERT Fine-tuning**: Measures perplexity of model trained on new language.
- **Vocabulary Size**: Total lexicon count.
- **Phonemic Inventory Size**: Number of phonemes used.
- **Phonotactic Complexity**: Assesses rule complexity.

Zipf's Law Metric

- Evaluates how closely the generated language follows Zipf's Law.
- Zipf exponent compared against that of the original English text.
- Indicator of natural distribution of word frequencies.

System Design and Implementation

Pipeline Overview

- Modular architecture for conlang generation.
- Core object: `LanguageDescription`.
- Pipeline managed by subclassing `Pipeline`.
- JSON-based output for module results.

Module Design

- Each module subclasses `Module`.
- Implements `execute()` method.
- Can add/modify language features.
- Checks dependencies and raises errors if needed.

Phonetics and Phonotactics

Phonetics Module

- Based on PHOIBLE phoneme segments.
- `MostCommonPhonemesModule`.

Phonotactics Module

- `BasicPhonotacticsModule` for C/V structures.
- `CustomPhonotacticsModule` supports constraints.

Grammar Modules

- Features via `GrammaticalFeatures` class.
- Uses `AbstractFeature` and `AbstractPOS`.
- Modules:
  - `BaselineAgglutinativeGrammarModule`
  - `BaselineIsolatingGrammarModule`

Vocabulary Modules

- Data in `VocabDictionary`.
- Generation:
  - `FromSourceVocabularyModule`
  - `ClusterTwoLevelVocabularyModule`
  - `ClusterAndSimplifyVocabularyModule`
  - `ApproximatingVocabularyModule`
- Mapping:
  - `RandomMappingModule`

Translation Module

- Translates using LLMs.
- Paragraph-level translation of `AbstractSourceText`.

Evaluation Framework

- Evaluations handled by `Evaluator` class.
- Independent of the main pipeline.
- Outputs saved in an `evaluations` folder.

Evaluation Modules

- **Translation Quality:**
    - `DetranslationEvaluator` (BLEU, METEOR, ROUGE)
    - `GenerateTranslationSummary`
- **Compression and Comprehension:**
    - `CompressionEvaluator`
    - `RaceCEvaluator`
- **Learnability and Naturalness:**
    - `BertEvaluator`
    - `ZipfsLawEvaluator`

Pipeline Variants

- **Baseline:** Most common phonemes, CV rules.
- **Approximation:** Fixed vocabulary + embedding-based matching.
- **Two-Level:** Bi-syllabic clustering-based vocabulary.
- **Cluster & Simplify:** One word per cluster.
- **Phonotactics:** Custom syllable constraints.

LLMs and Embedding Models

- **Local LLMs:** via Ollama, e.g. `deepseek-r1:14b`.
- **Remote LLMs:** OpenAI and Groq (`gpt-4o`).
- **Embeddings:** `all-MiniLM-L6-v2`.

Clustering Methods

- Subclasses of `ClusteringMethod`.
- Vocabulary clustering: `AgglomerativeClustering`.
- Alternatives supported: KMeans, DBScan, HDBScan.

Datasets and Tools

- Dataset: `RaceC` (Race-C dataset).
- Word frequency: `wordfreq` library.
- Corpus source: Exquisite Corpus.
- Outputs: HTML summaries and translation pairs.

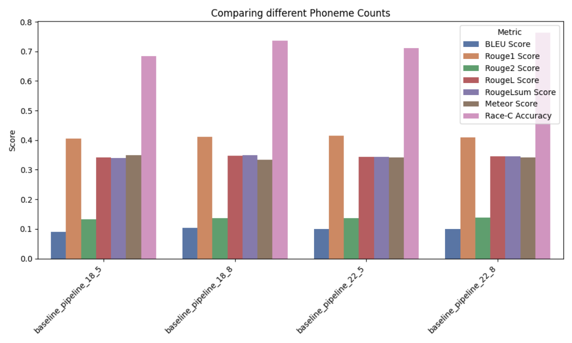
Results and Evaluation

Overview

- Discusses key experimental results.
- Evaluates effects of parameters on performance.
- Summarizes findings.
- Outlines directions for future work.

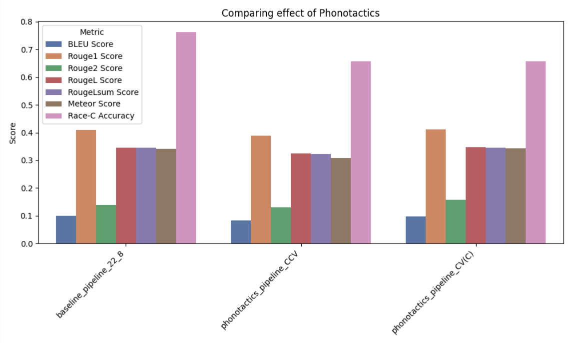
Effect of Phoneme Count

- Minimal impact on translation or Race-C scores.
- Suggests that languages can be simplified in phoneme inventory.
- Meaning preservation not significantly affected.



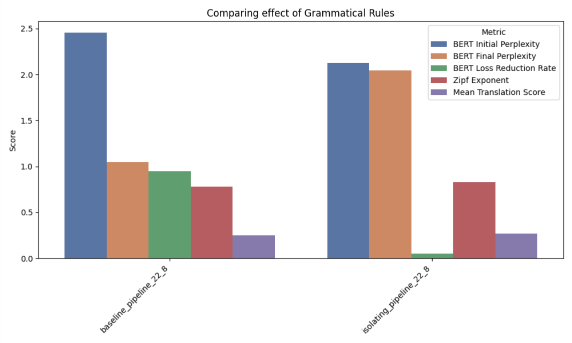
Effect of Phonotactics

- Simplified phonotactic rules do not degrade performance.
- Translation and comprehension scores largely unaffected.
- Simplification viable for phonological structure.



Effect of Grammar Rules

- Compared Agglutinative vs. Isolating grammars.
- No major difference in translation or comprehension.
- Only Bert perplexity varied—needs further analysis.



Vocabulary Simplification and Performance

- Vocabulary simplification causes performance drop.
- MTS (n-gram) suffers more than Race-C (meaning-based).
- Ratio of score to vocab size is better for simpler vocab methods.

Effect of Simplified Vocabulary

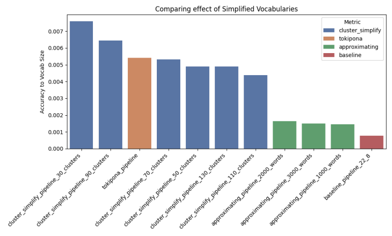


Figure 1: Race-C Accuracy

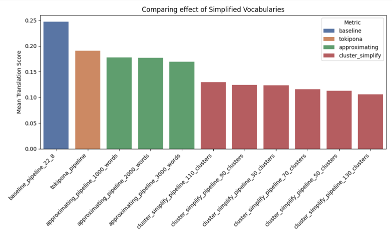
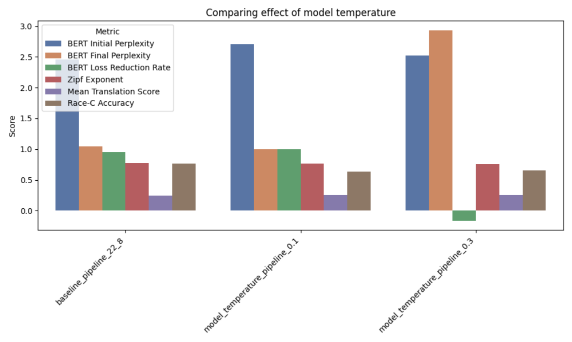


Figure 2: Mean Translation Score (MTS)

Effect of Language Model Temperature

- High temperature leads to less consistent results.
- Low to moderate values (e.g., 0.2) are stable.
- Reproducibility improves with lower temperature.



Conclusion

Conclusion

- LLMs can aid in building efficient, minimal conlangs.
- Simplification is possible in phonology, grammar, and vocab.
- Reading comprehension is robust even with simplification.
- Zipf-like distributions emerged naturally.
- Framework enables modular experimentation and evaluation.

Future Work

- Explore more grammar/vocab generation strategies.
- Better metrics for simplicity and efficiency.
- Deeper analysis of grammatical influence on meaning retention.
- Extend to spoken or multi-modal languages.

Questions?

Thank you!

Questions are welcome.