



Paper Type: Original Article

Metaheuristic Optimization of Prompt Structures for Low-Resource Language Models

Abdulrahman Al-Harthy^{1,*}, Nguyen Thi Lan²

¹ Department of Computer Science, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia; a.harthy@kau.edu.sa.

² School of Information and Communication Technology, Hanoi University of Science and Technology, Hanoi, Vietnam; lan.nguyen@hust.edu.vn.

Citation:

Received: 22 February 20206

Revised: 10 May 2026

Accepted: 08 June 2026

Al-Harthy, A., & Lan, N. T. (2025). Metaheuristic optimization of prompt structures for low-resource language models. *Metaheuristic Algorithms with Applications*, 3(3), 316-340.

Abstract

Large Language Models (LLMs) have demonstrated remarkable capabilities across a range of Natural Language Processing (NLP) tasks when guided by carefully designed prompts. However, their performance degrades substantially on low-resource languages, where limited pretraining data and the absence of large-scale labeled corpora render conventional prompt engineering strategies ineffective. In this paper, we introduce Evolutionary Prompt Optimization for Low-Resource Language Models (EvoPrompt-LR), a novel framework that employs a hybrid Genetic Algorithm with Simulated Annealing (GA-SA) to automatically discover optimal prompt structures for low-resource languages. EvoPrompt-LR features four key innovations: 1) a hierarchical prompt genome encoding that jointly optimizes instruction templates, in-context example selection and ordering, Chain-of-Thought (CoT) scaffold structures, and output format specifications, 2) a cross-lingual fitness transfer mechanism that evaluates prompt candidates on high-resource proxy languages before selectively fine-evaluating top candidates on the target low-resource language, reducing expensive evaluations by approximately 80%, 3) semantic diversity preservation via embedding-space crowding distance to maintain a diverse population of prompt candidates, and 4) language-specific mutation operators that respect the morphological and syntactic constraints of typologically diverse languages, including agglutinative (Vietnamese, Turkish, Uzbek), fusional (Arabic), and isolating (Khmer) language families. We evaluate EvoPrompt-LR on six low-resource languages: Arabic (dialectal), Vietnamese, Swahili, Yoruba, and Khmer, across four NLP tasks: Sentiment analysis, Named Entity Recognition (NER), Question Answering (QA), and text summarization, using LLaMA-3-8B, Mistral-7B, and AceGPT-7B as base models. Experimental results demonstrate that EvoPrompt-LR achieves improvements of 8–22% in accuracy and F1 score over the strongest baselines, with the largest gains observed on truly low-resource languages (Yoruba: +22%, Khmer: +19%). On four of six languages, EvoPrompt-LR attains performance competitive with Supervised Fine-Tuning (SFT) approaches that use 100× more labeled data.

Keywords: Prompt optimization, Low-resource languages, Metaheuristic, Genetic algorithm, Large language models, Natural language processing, Cross-lingual transfer, Few-shot learning.

1 | Introduction

The advent of Large Language Models (LLMs) has fundamentally transformed the landscape of Natural Language Processing (NLP). Models such as GPT-3 [1], LLaMA [2], LLaMA-2 [3], LLaMA-3 [4], and Mistral [5] have achieved state-of-the-art results across diverse benchmarks by leveraging vast corpora during

pretraining and demonstrating emergent abilities in few-shot and zero-shot settings. A critical determinant of LLM performance is the prompt, the natural language instruction or context provided to the model to elicit a desired response. The field of prompt engineering has consequently emerged as a central research direction, with studies demonstrating that subtle variations in prompt wording, structure, and composition can yield performance differences of 20% or more on the same task [6], [7].

Concurrently, research on Chain-of-Thought (CoT) prompting [8] has revealed that instructing LLMs to articulate intermediate reasoning steps dramatically improves performance on complex reasoning tasks. Extensions such as self-consistency [9], tree-of-thought [10], and automatic CoT [11] have further expanded the prompting toolkit. These advances, however, have been developed and validated predominantly in English, with limited investigation into their transferability to other languages, particularly those with scarce digital resources.

The challenge of low-resource languages represents one of the most pressing problems in contemporary NLP. Joshi et al. [12] documented the stark disparity in linguistic diversity within the NLP community, classifying the world's approximately 7,000 languages into six categories based on resource availability, with the vast majority falling into the lowest tiers. For languages such as Yoruba, Khmer, and Uzbek, spoken by tens of millions of people, the available pretraining corpora are orders of magnitude smaller than those for English or Chinese, resulting in degraded LLM performance characterized by increased hallucination, poor instruction following, and limited domain knowledge [13], [14]. While cross-lingual models such as XLM-R [15] and mBERT [16] have demonstrated transfer learning capabilities, their performance on genuinely low-resource languages remains substantially below that achieved on high-resource counterparts [17].

In response to the labor-intensive nature of manual prompt design, automated prompt optimization methods have recently attracted significant attention. Automatic Prompt Engineer (APE) [18] leverages LLMs themselves to generate and select task instructions. Optimization by Prompting (OPRO) [19] frames prompt optimization as a meta-optimization problem where the LLM iteratively refines its own instructions. Evolutionary Prompt Optimization (EvoPrompt) [20] connects Evolutionary Algorithms (EAs) with LLMs to perform discrete prompt optimization, demonstrating improvements of up to 25% on Big-Bench Hard tasks. Despite their effectiveness, these methods share a critical limitation: They assume access to substantial evaluation data in the target language and have been designed and tested almost exclusively on high-resource languages. When applied to low-resource settings, these approaches face three compounding challenges: 1) The search space of possible prompts is effectively unbounded and language-dependent, 2) Evaluation of prompt candidates requires labeled data that is scarce by definition, and 3) the LLMs themselves have poor linguistic competence in the target language, making prompt sensitivity even more acute.

This paper addresses these challenges through Evolutionary Prompt Optimization for Low-Resource Language Models (EvoPrompt-LR), a metaheuristic framework that systematically discovers effective prompt structures for low-resource languages. Our approach makes four principal contributions:

- I. Hierarchical prompt genome encoding: We introduce a four-level genome representation that decomposes prompts into independently evolvable components instruction templates, in-context example configurations, CoT scaffolds, and output format specifications enabling fine-grained optimization of each structural element.
- II. Cross-lingual fitness transfer: We propose a mechanism that evaluates prompt candidates first on a high-resource proxy language using translation-based approximate fitness, selectively promoting top candidates for evaluation on the actual low-resource target language, thereby reducing expensive low-resource evaluations by approximately 80%.
- III. Semantic diversity preservation: We adapt the concept of crowding distance from multi-objective evolutionary optimization [21] to the embedding space of prompts, maintaining a diverse population that avoids premature convergence to locally optimal prompt structures.

- IV. Language-specific mutation operators: We design mutation operators that respect the morphological and syntactic constraints of different language typologies, including agglutinative, fusional, and isolating languages, ensuring that evolved prompts remain linguistically valid and natural.

We evaluate EvoPrompt-LR across six low-resource languages: Arabic (dialectal varieties: Gulf, Egyptian, and Maghrebi), Vietnamese, Swahili, Yoruba, Uzbek, and Khmer, on four canonical NLP tasks: Sentiment Analysis, Named Entity Recognition (NER), Question Answering (QA), and text summarization. Experiments with three base models (LLaMA-3-8B, Mistral-7B, and AceGPT-7B) demonstrate consistent improvements of 8–22% in accuracy and F1 score over the strongest baselines, with the largest gains on the most resource-constrained languages. Notably, on four of six languages, EvoPrompt-LR achieves performance competitive with Supervised Fine-Tuning (SFT) methods that require 100× more labeled data, underscoring the potential of intelligent prompt optimization as a cost-effective alternative to data-intensive approaches for low-resource NLP.

2 | Related Work

2.1 | Prompt Engineering and Optimization

The importance of prompt design for LLM performance has been extensively documented since the introduction of GPT-3 [1], which demonstrated that few-shot prompting could rival fine-tuning on many benchmarks. Subsequent work formalized the study of prompt sensitivity, showing that even semantically equivalent reformulations of prompts could produce dramatically different results [7], [22]. This observation motivated a transition from manual prompt crafting toward systematic optimization approaches.

Early automated methods include prefix tuning [23] and prompt tuning [24], which optimize continuous prompt embeddings via gradient descent. While effective, these soft-prompting methods require access to model parameters and produce non-interpretable prompts, limiting their applicability to black-box API settings and their transferability across models. In contrast, discrete prompt optimization methods operate directly on natural language tokens. APE [18] uses an LLM to generate candidate instructions given input-output demonstrations and selects the highest-performing one. OPRO [19] maintains an optimization trajectory of prompt-score pairs and instructs the LLM to propose improved prompts based on this history. ProTeGi [25] employs gradient-inspired textual feedback to iteratively refine prompts. PromptBreeder [26] introduces a self-referential evolutionary approach where mutation operators themselves evolve alongside the prompts they generate.

Despite these advances, existing prompt optimization methods overwhelmingly target English-language tasks and assume the availability of substantial development sets for evaluation. Our work extends this line of research to the low-resource setting, where both assumptions are violated.

2.2 | Low-Resource Natural Language Processing

The disparity in NLP resources across languages has been a longstanding concern. Joshi et al. [12] categorized the world's languages into six classes based on available resources and community engagement, revealing that the vast majority of languages, including many with millions of speakers, fall into the lowest resource tiers. This taxonomy has served as a foundational reference for subsequent low-resource NLP research.

Traditional approaches to low-resource NLP have focused on data augmentation [27], cross-lingual transfer from high-resource languages [28], [29], and multilingual pretraining [15], [30]. More recently, the emergence of multilingual LLMs has provided new avenues for low-resource language processing. Studies have examined the multilingual capabilities of models such as GPT-4 [31], BLOOM [32], and LLaMA-2 [3], finding that while these models exhibit some cross-lingual transfer, their performance on low-resource languages remains significantly below their English-language capabilities [13], [14], [33].

Specialized models targeting specific low-resource languages have also been developed, including AceGPT for Arabic [34], SEA-LION for Southeast Asian languages [35], and AfroLM for African languages [36].

While these models improve baseline performance, they require substantial resources for training and do not address the fundamental challenge of prompt optimization in data-scarce settings.

2.3 | Evolutionary and Metaheuristic Approaches for Natural Language Processing

EAs have a long history of application to NLP tasks, predating the current LLM era. Genetic Algorithms (GAs) [37] have been applied to feature selection for text classification [38], grammar optimization [39], and information extraction rule learning [40]. The seminal work of Goldberg [37] established the theoretical foundations for GAs, including schema theory and the building block hypothesis, which inform our hierarchical genome design.

Simulated Annealing (SA) [41] has been employed for combinatorial optimization in NLP contexts, including word sense disambiguation [42], text summarization [43], and machine translation decoding [44]. The combination of GA and SA, hybridizing the population-based exploration of GAs with the controlled acceptance of inferior solutions characteristic of SA, has shown particular promise for avoiding local optima in rugged fitness landscapes [45], [46].

Most recently, EvoPrompt [20] demonstrated the effectiveness of connecting EAs with LLMs for prompt optimization, achieving state-of-the-art results across 31 datasets. However, EvoPrompt operates exclusively on instruction-level prompts in English and does not address the hierarchical structure of modern prompts (which include examples, reasoning scaffolds, and format specifications) or the unique challenges of low-resource settings. Our work substantially extends this paradigm.

2.4 | Cross-Lingual Transfer

Cross-lingual transfer learning aims to leverage resources from high-resource languages to improve performance on low-resource targets. Foundational work includes multilingual word embeddings [47], cross-lingual model adaptation [48], and zero-shot cross-lingual classification [15]. In the context of LLMs, studies have explored cross-lingual prompting strategies, where prompts in one language are used to elicit responses in another [49–51].

Particularly relevant to our work, Shi et al. [51] demonstrated that CoT reasoning can be elicited in multilingual settings by providing English-language CoT prompts even when the input is in another language, suggesting that prompt structure transfers across languages more readily than prompt content. Winata et al. [52] examined cross-lingual in-context learning, finding that the language of demonstrations can be decoupled from the language of the query with varying degrees of success depending on language similarity and model architecture. Our cross-lingual fitness transfer mechanism builds upon these findings by systematically exploiting prompt-structure transferability to reduce the evaluation burden on low-resource languages.

3 | Methodology

This section presents the EvoPrompt-LR framework in detail. We begin with the formal problem formulation (Section 3.1), then describe the hierarchical prompt genome encoding (Section 3.2), the hybrid Genetic Algorithm with Simulated Annealing (GA-SA) optimization algorithm (Section 3.3), the cross-lingual fitness transfer mechanism (Section 3.4), language-specific mutation operators (Section 3.5), and the semantic diversity preservation strategy (Section 3.6).

3.1 | Problem Formulation

Let M denote a pretrained language model and $T = \{(x_i, y_i)\}_{i=1}^n$ denote a small labeled dataset in a low-resource target language L_t , where n is typically in the range of 50–200 examples. A prompt P is a structured natural language input that, when concatenated with a test instance x , elicits a response $\hat{y} = M(P \oplus x)$ from the model. The objective is to find the optimal prompt P^* that maximizes task performance:

$$P^* = \arg \max_{P \in \mathcal{P}} \mathbb{E}_{(x,y) \sim T} [f(M(P \oplus x), y)], \quad (1)$$

where $\mathcal{P}$ denotes the space of all valid prompts, $f(0,0)$ is a task-specific evaluation metric (e.g., accuracy, F1 score, ROUGE), and $\oplus$ denotes the concatenation operation. The key challenge in the low-resource setting is that $\mathcal{P}$ is combinatorially large, n is small (making fitness estimates noisy), and the model M has limited competence in L_e .

We decompose the prompt P into four structural components:

$$P = (P^{\text{inst}}, P^{\text{demo}}, P^{\text{cot}}, P^{\text{fmt}}), \quad (2)$$

where P^{inst} is the instruction template (task description), P^{demo} is the in-context demonstration configuration (selection and ordering of K examples from a seed pool S), P^{cot} is the CoT scaffold (reasoning template with variable decomposition depth), and P^{fmt} is the output format specification. This decomposition enables independent optimization of each component while preserving the interdependencies through the fitness evaluation.

3.2 | Hierarchical Prompt Genome Encoding

We represent each prompt candidate as a four-level hierarchical genome $G = (g_1, g_2, g_3, g_4)$, where each level corresponds to one of the four structural components defined in Eq. (2).

Level 1 (instruction template gene (g_1)). The instruction template gene encodes the task description using a combinatorial template grammar. We define a context-free grammar $G_{\text{inst}} = (V, \Sigma, R, S)$ where V is the set of non-terminal symbols representing structural slots (e.g., $\langle \text{task_verb} \rangle$, $\langle \text{domain_context} \rangle$, $\langle \text{constraint} \rangle$), Σ is the set of terminal symbols (language-specific lexical items), R is the set of production rules, and S is the start symbol. The gene g_1 is a vector of production rule indices:

$$g_1 = (r_1, r_2, \dots, r_m), \text{ where } r_j \in R(v_j). \quad (3)$$

This encoding ensures that all generated instructions are syntactically valid while enabling combinatorial exploration of phrasing alternatives. For example, the non-terminal $\langle \text{task_verb} \rangle$ for sentiment analysis may expand to (classify, determine, identify, analyze, assess), producing semantically similar but structurally distinct instruction variants.

Level 2 (demonstration configuration gene (g_2)). The demonstration gene encodes which K examples are selected from a seed pool $S = \{s_1, \dots, s_N\}$ and in what order they are presented. We represent this as a permutation-selection pair:

$$g_2 = (\sigma, K), \text{ where } \sigma \in \text{Perm}(S), K \in \{0, 1, \dots, K_{\text{max}}\}. \quad (4)$$

The first K elements of the permutation σ are selected as in-context demonstrations, preserving their order. This encoding naturally captures both selection and ordering decisions in a single genetic representation. We set $K_{\text{max}} = 8$ in our experiments, consistent with the context length constraints of the base models.

Level 3 (CoT scaffold gene (g_3)). The CoT scaffold gene encodes a reasoning template with variable decomposition depth. We represent reasoning chains as Directed Acyclic Graphs (DAGs) where nodes correspond to reasoning steps and edges encode dependencies:

$$g_3 = (D, \{t_d\}_{d=1}^D, E), \text{ where } D \in \{0, 1, \dots, D_{\text{max}}\}. \quad (5)$$

Here, D is the decomposition depth (number of explicit reasoning steps), t_d is the template for step d (drawn from a predefined set of reasoning templates such as "First, identify the key entities," "Then, analyze the relationship between," "Consider the context of"), and $E \subseteq \{1, \dots, D\} \times \{1, \dots, D\}$ is the set of dependency edges. Setting $D = 0$ corresponds to direct (non-CoT) prompting. We set $D_{\text{max}} = 5$ based on preliminary experiments.

Level 4 (output format gene (g4)). The output format gene specifies the expected response structure:

$$g_4 \in \{\text{JSON, NL, CSV, XML, Label-Only}\}, \quad (6)$$

where JSON denotes structured JSON output, NL denotes free-form natural language, and Label-Only denotes single-token classification labels. The format specification is prepended to the prompt as an explicit output instruction (e.g., "Respond with a JSON object containing the fields: Sentiment, confidence").

3.3 | Genetic Algorithm with Simulated Annealing Hybrid Optimization Algorithm

The core of EvoPrompt-LR is a hybrid algorithm that combines the population-based search of GAs [37] with the controlled stochastic acceptance of SA [41]. The GA component maintains a diverse population of prompt candidates and applies genetic operators (selection, crossover, mutation) to explore the search space, while the SA component governs the acceptance of offspring into the population, allowing temporary acceptance of inferior solutions to escape local optima.

The fitness of a prompt genome G is defined as

$$F(G) = \alpha \cdot f_{\text{task}}(G) + \beta \cdot f_{\text{robust}}(G) + \gamma \cdot f_{\text{length}}(G), \quad (7)$$

where f_{task} is the primary task metric (accuracy or F1), f_{robust} measures performance stability across multiple evaluation runs (inverse of standard deviation), f_{length} penalizes excessively long prompts to remain within context length budgets, and α, β, γ are weighting hyperparameters with $\alpha = 0.8, \beta = 0.15, \gamma = 0.05$ in our experiments.

Selection: We employ tournament selection with tournament size $k = 3$. In each tournament, k individuals are randomly sampled from the population, and the individual with the highest fitness is selected as a parent.

Crossover: Given the hierarchical genome structure, we implement level-aware crossover. For two parent genomes G_A and G_B , the offspring G_C is produced by independently selecting each level from one of the parents:

$$g_i^C = g_i^A \text{ with probability } p_{\text{cross}}, \quad g_i^B \text{ otherwise} \quad \text{for } i \in \{1, 2, 3, 4\}, \quad (8)$$

where $p_{\text{cross}} = 0.5$ by default. Additionally, within *Level 1*, we implement single-point crossover on the production rule vector, and within *Level 2*, we implement Order Crossover (OX) on the permutation component to preserve relative ordering information [53].

Mutation: Each genome level is mutated independently with level-specific mutation rates $\mu_1, \mu_2, \mu_3, \mu_4$. For *Level 1*, mutation replaces a randomly selected production rule with an alternative from the same non-terminal. For *Level 2*, mutation performs either a swap (exchanging two examples), an insertion (adding a new example from the pool), or a deletion (removing an example). For *Level 3*, mutation adjusts the decomposition depth or replaces a step template. For *Level 4*, mutation switches to an alternative output format. Language-specific mutation operators (Section 3.5) are applied at *Level 1* and *Level 3* to ensure linguistic validity.

SA-based acceptance: After generating offspring via crossover and mutation, the acceptance into the next generation is governed by the Metropolis criterion:

$$P_{\text{accept}}(G_{\text{new}}) = \min(1, \exp((F(G_{\text{new}}) - F(G_{\text{old}})) / T)), \quad (9)$$

where T is the temperature parameter that decreases according to a geometric cooling schedule $T_{t+1} = \eta \cdot T_t$ with cooling rate $\eta = 0.95$. The initial temperature T_0 is calibrated so that solutions with fitness 5% below the current best are accepted with probability 0.5 in the first generation.

The complete hybrid algorithm is presented in *Algorithm 1*.

Algorithm 1. EvoPrompt-LR main loop.

```

Input:
Seed pool
S
, evaluation set
T
, model
M
, population size
N
pop
,
max generations
G
max
, cooling rate  $\eta$ , proxy language
L
p
Output:
Optimal prompt genome
G
*
1:
Pop
 $\leftarrow$  InitializePopulation(
N
pop
,
S
)
// Random genome initialization
2:
T
0
 $\leftarrow$  CalibrateTemperature(
Pop
,
T
,
M
)
3:
for
t
= 1
to
G
max
do
4:
T
t
 $\leftarrow \eta$ 
t
•
T
0
5:
Offspring
 $\leftarrow \emptyset$ 
6:
for
j

```

```

= 1
to
N
pop
/ 2
do
7:
G
A
,
G
B
← TournamentSelect(
Pop
,
k
=3)
8:
G
C
,
G
D
← HierarchicalCrossover(
G
A
,
G
B
)
9:
G
C
← LanguageAwareMutate(
G
C
,
L
t
)
10:
G
D
← LanguageAwareMutate(
G
D
,
L
t
)
11:
Offspring
←
Offspring
U {
G
C
,
G
D
}
12:
end for

```

```

13:
  Offspring
  ← CrossLingualFitnessTransfer(
    Offspring
    ,
    L
    P
    ,
    L
    t
    ,
    M
  )
  // Alg. 2
14:
  for each
    G
    new
    ∈
    Offspring
  do
    15:
      G
      old
      ← RandomSelect(
        Pop
      )
    16:
      if
        F
        (
          G
          new
        )
        >
        F
        (
          G
          old
        )
      or
        rand()
        <
        exp( $\Delta$ 
          F
          /
          T
          t
        )
      then
    17:
      Replace
      G
      old
      with
      G
      new
      in
      Pop
    18:
      end if
    19:
      end for

```

```

20:
Pop
← DiversityPreservation(
Pop
)
// Crowding distance (Sec. 3.6)
21:
UpdateElite(
G
*)
22:
end for
23:
return
G
*

```

3.4 | Cross-Lingual Fitness Transfer Mechanism

A central challenge in optimizing prompts for low-resource languages is the cost of fitness evaluation: Each candidate prompt must be applied to multiple test instances in the target language, consuming both computational resources and scarce labeled data. We address this through a cross-lingual fitness transfer mechanism that exploits the observation that prompt structure transfers across languages more robustly than prompt content [51], [52].

Let L_p denote a high-resource proxy language (English or Chinese in our experiments) with an abundant evaluation set T_p . For a candidate genome G , we define the proxy fitness as

$$\tilde{F}(G) = F(\text{Translate}(G, L_t \rightarrow L_p); T_p, M), \quad (10)$$

where $\text{Translate}(G, L_t \rightarrow L_p)$ translates the textual components of the genome (instruction template, CoT steps, format specifications) from the target language to the proxy language while preserving the structural components (example selection indices, ordering, depth parameters). Translation is performed using NLLB-200 [54] for all language pairs.

The cross-lingual fitness transfer operates in two phases, as detailed in *Algorithm 2*.

Algorithm 2. Cross-lingual fitness transfer.

```

Input:
Offspring set
O
, proxy language
L
P
, target language
L
t
,
model
M
, top-
k
ratio
r
= 0.2
Output:
Offspring with fitness scores
// Phase 1: Proxy evaluation (all candidates)
1:
for each

```

```

G
∈
O
do
2:
G
proxy
← Translate(
G
,
L
t
→
L
P
)
3:
F□
(
G
) ← Evaluate(
G
proxy
,
T
P
,
M
)
4:
end for
// Phase 2: Target evaluation (top-k candidates only)
5:
O
top
← Top-[
r
· |
O
|| by
F□
6:
for each
G
∈
O
top
do
7:
F
(
G
) ← Evaluate(
G
,
T
,
M
)
// Direct low-resource evaluation
8:
end for
// Assign estimated fitness to remaining candidates

```

```

9:
for each
G
 $\in$ 
O
\
O
top
do
10:
F
(
G
)  $\leftarrow \lambda \cdot$ 
F□
(
G
)
// Discounted proxy fitness,  $\lambda = 0.85$ 
11:
end for
12:
return
O
with fitness scores

```

By evaluating only the top 20% of candidates on the actual low-resource language, we reduce the number of expensive low-resource evaluations by approximately 80% per generation. The discount factor $\lambda = 0.85$ accounts for the imperfect correlation between proxy and target fitness, ensuring that candidates not directly evaluated on the target language are not overvalued.

To calibrate the proxy-target correlation, we perform a one-time calibration step at initialization where a subset of 30 randomly generated prompts are evaluated on both L_p and L_t . We compute the Spearman rank correlation ρ between the two fitness rankings and adjust the discount factor accordingly: $\lambda = 0.7 + 0.2 \cdot \rho$. In our experiments, ρ ranges from 0.62 (Khmer–English) to 0.81 (Arabic–English), reflecting the varying degrees of structural similarity between proxy and target languages.

3.5 | Language-Specific Mutation Operators

A distinctive feature of EvoPrompt-LR is the incorporation of mutation operators tailored to the typological characteristics of the target language. Standard mutation operators that randomly substitute words or phrases are linguistically naïve and frequently produce ungrammatical or semantically incoherent prompts, particularly in morphologically rich languages. We design three families of mutation operators corresponding to major typological classes:

Agglutinative language operators (Vietnamese, Uzbek): In agglutinative languages, words are formed by concatenating morphemes, each carrying a single meaning or grammatical function. Mutation operators for these languages operate at the morpheme level rather than the word level. Specifically, our morpheme-aware substitution operator identifies the morphological decomposition of instruction keywords using the UniMorph database [55] and substitutes individual morphemes while preserving the overall morphological structure. For example, in Uzbek, the verb stem *yoz-* (write) may be mutated to *belgi-* (mark) while preserving the suffix chain *-ish-ing-iz* (reciprocal-present-2PL.formal). For Vietnamese, which is analytic but uses classifier systems and serial verb constructions, our operator respects classifier-noun agreement and serial verb ordering constraints.

Fusional language operators (Arabic): Arabic exhibits root-and-pattern morphology where consonantal roots are interleaved with vocalic patterns to produce word forms. Our root-preserving mutation operator

maintains the triconsonantal root while varying the morphological pattern. For instance, the root k-t-b (related to writing) may be realized as kataba (he wrote), kitāb (book), kātib (writer), or maktūb (written), allowing semantically related variations that remain morphologically well-formed. Additionally, our operator handles Arabic dialectal variation by maintaining a mapping between Modern Standard Arabic (MSA) forms and their dialectal equivalents for Gulf, Egyptian, and Maghrebi Arabic.

Isolating language operators (Khmer): As a predominantly isolating language, Khmer has minimal morphological inflection but exhibits complex syntactic constraints, including specific word ordering, particle use, and register-dependent vocabulary. Our operator for Khmer focuses on syntactic slot permutation (reordering of adverbial modifiers, topic-comment restructuring) and register-appropriate lexical substitution, drawing on a curated register lexicon of formal/informal variant pairs.

For Swahili and Yoruba, which exhibit elements of agglutination and tonal marking respectively, we implement hybrid operators that combine morpheme-level substitution with tone-preserving constraints (for Yoruba) and noun class agreement preservation (for Swahili).

3.6 | Semantic Diversity Preservation

To prevent premature convergence to a narrow region of the prompt space, a common failure mode in evolutionary optimization, we incorporate a semantic diversity preservation mechanism inspired by the crowding distance used in NSGA-II [21].

We compute the semantic embedding of each prompt genome G by encoding its rendered prompt text using a multilingual sentence encoder (LaBSE) [56]. Let $e(G) \in \mathbb{R}^d$ denote the embedding. For each individual in the population, the crowding distance is computed as

$$CD(G_i) = \sum_{j \in NN_k(i)} \|e(G_i) - e(G_j)\|_2, \quad (11)$$

where $NN_k(i)$ denotes the k -nearest neighbors of individual i in the embedding space (we use $k = 3$). During population trimming (when the combined parent-offspring population exceeds N_{pop}), individuals with lower fitness are eliminated, but ties are broken in favor of higher crowding distance, preserving diverse prompt structures.

Additionally, we implement a diversity injection mechanism: Every δ generations (we use $\delta = 10$), we replace the bottom 10% of the population (by fitness) with freshly generated random genomes. This mechanism ensures continuous exploration of unexplored regions of the prompt space and has been shown to improve long-run convergence in similar evolutionary optimization contexts [57].

4 | Experimental Setup

4.1 | Languages and Datasets

We evaluate EvoPrompt-LR on six languages selected to represent diverse typological families, geographic regions, and resource availability levels, following the taxonomy of Joshi et al. [12]. *Table 1* summarizes the language characteristics and datasets used.

Table 1. Languages, typological characteristics, and datasets used in experiments.

Language	Family	Typology	Resource Level	Sentiment	NER	QA	Summarization
Arabic (Dialectal)	Afroasiatic	Fusional	Mid-Low	ArSenTD [58]	ANERcorp [59]	TyDi QA [60]	XL-Sum [61]
Vietnamese	Austroasiatic	Isolating/Analytic	Mid-Low	UIT-VSFC [62]	PhoNER [63]	UIT-ViQuAD [64]	XL-Sum [61]
Swahili	Niger–Congo (Bantu)	Agglutinative	Low	AfriSenti [65]	MasakhaNER [66]	AfriQA [67]	XL-Sum [61]

Yoruba	Niger–Congo (Volta–Niger)	Isolating/Tonal	Very Low	AfriSenti [65]	MasakhaNER [66]	AfriQA [67]	XL-Sum [61]
Uzbek	Turkic	Agglutinative	Very Low	UzSentiment [68]	UzNER (custom, 500 annot.)	UzQA (custom, 400 pairs)	XL-Sum [61]
Khmer	Austroasiatic	Analytic/Isolating	Very Low	KhmerSenti (custom, 600 annot.)	KhmerNER (custom, 450 annot.)	KhmerQA (custom, 350 pairs)	XL-Sum [61]

For the seed pool used in prompt optimization, we sample 100 labeled examples per task per language from the respective training sets. The evaluation is performed on held-out test sets of 500 examples (or the full test set if smaller). For languages where existing benchmarks are unavailable (Uzbek NER, Uzbek QA, Khmer sentiment, Khmer NER, Khmer QA), we construct custom datasets through expert annotation, following established annotation guidelines [65], [66]. Custom datasets undergo two rounds of annotation by native speakers with inter-annotator agreement (Cohen's κ) exceeding 0.80 for all tasks.

4.2 | Base Models

We evaluate EvoPrompt-LR with three base LLMs chosen to represent different model families and linguistic specializations:

- I. LLaMA-3-8B [4]: A general-purpose open-source model with 8 billion parameters, pretrained on over 15 trillion tokens of multilingual data. While primarily English-centric, LLaMA-3 includes non-trivial coverage of Arabic, Vietnamese, and Swahili.
- II. Mistral-7B [5]: A 7-billion-parameter model employing grouped-query attention and sliding window attention mechanisms. Known for strong performance-to-parameter efficiency, it serves as a representative of efficient LLM architectures.
- III. AceGPT-7B [34]: A 7-billion-parameter model specifically fine-tuned for Arabic using continued pretraining on Arabic corpora and reinforcement learning with culturally aligned feedback. Included to assess the interaction between language-specialized models and prompt optimization.

All models are used in their instruction-tuned variants. Inference is performed with greedy decoding (temperature = 0) for classification tasks and nucleus sampling (top-p = 0.9, temperature = 0.3) for generation tasks. The maximum output length is set to 512 tokens for QA and summarization tasks and 64 tokens for sentiment and NER tasks.

4.3 | Baselines

We compare EvoPrompt-LR against six baseline methods spanning manual and automated prompt optimization approaches:

- I. Zero-shot: Direct task instruction without demonstrations, using the best manually crafted prompt identified through preliminary experiments.
- II. Few-shot (Standard): Fixed set of $K = 5$ randomly selected demonstrations prepended to the task instruction, using the same instruction as zero-shot.
- III. Manual prompting: Task-specific prompts hand-crafted by native speakers of each target language with NLP expertise, following best practices from the prompt engineering literature [6], [69].
- IV. Automatic prompt engineer [18]: APE, which generates candidate instructions via an LLM and selects the best-performing one. We use the iterative refinement variant.
- V. Optimization by prompting [19]: OPRO, which uses an LLM to iteratively propose improved instructions based on an optimization trajectory. We use LLaMA-3-8B as the optimizer LLM.

- VI. Evolutionary Prompt Optimization [20]: EvoPrompt uses GA and differential evolution operators applied to instruction-level prompts, as described in the original implementation.

Additionally, we compare against SFT using LoRA [70] with the full available training data for each dataset (typically 5,000–50,000 labeled examples) to establish an upper bound reference.

4.4 | Evaluation Metrics

We adopt task-specific evaluation metrics consistent with the conventions of each task:

- I. Sentiment analysis: Macro-averaged F1 score.
- II. NER: Entity-level F1 score, following the CoNLL evaluation scheme.
- III. QA: Exact Match (EM) and token-level F1 score.
- IV. Text summarization: ROUGE-L F1 score [71] and BERTScore F1 [72].

All experiments are repeated with three random seeds, and we report the mean $\pm$ standard deviation. Statistical significance is assessed using the Wilcoxon signed-rank test at the $p < 0.05$ level.

4.5 | Implementation Details

EvoPrompt-LR is implemented in Python using the Hugging Face Transformers library [73]. Key hyperparameters for the evolutionary optimization are: Population size $N_{\text{pop}} = 50$, maximum generations $G_{\text{max}} = 100$, tournament size $k = 3$, crossover probability $p_{\text{cross}} = 0.5$, mutation rates $\mu_1 = 0.3$, $\mu_2 = 0.2$, $\mu_3 = 0.15$, $\mu_4 = 0.1$, cooling rate $\eta = 0.95$, proxy evaluation ratio $r = 0.2$, diversity injection interval $\delta = 10$. The template grammar for each language is constructed by two native-speaking NLP researchers and contains between 120 (Khmer) and 450 (Arabic) production rules.

Experiments are conducted on a cluster of 8 NVIDIA A100 80GB GPUs. A single optimization run (100 generations $\times$ 50 population size) requires approximately 12–18 hours of compute time depending on the language and task, compared to approximately 60–90 hours without the cross-lingual fitness transfer mechanism.

5 | Results

We present our experimental results across four NLP tasks and six languages. We first report the main comparative results (Section 5.1), followed by model-specific analysis (Section 5.2), ablation studies (Section 5.3), and analysis of convergence behavior and computational costs (Section 5.4).

5.1 | Main Results

Table 2 presents the sentiment analysis results across all six languages using LLaMA-3-8B as the base model. EvoPrompt-LR consistently achieves the highest F1 scores, with improvements ranging from 8.3% (Vietnamese) to 22.1% (Yoruba) over the best-performing baseline.

Table 2. Sentiment analysis results (Macro F1 %) for LLaMA-3-8B.

Method	Arabic (Dial.)	Vietnamese	Swahili	Yoruba	Uzbek	Khmer	Avg.
Zero-shot	41.2 $\pm$ 1.1	52.8 $\pm$ 0.9	35.4 $\pm$ 1.8	28.6 $\pm$ 2.3	31.7 $\pm$ 2.0	26.3 $\pm$ 2.5	36.0
Few-shot (K=5)	49.7 $\pm$ 1.4	58.3 $\pm$ 1.2	42.1 $\pm$ 1.5	34.9 $\pm$ 1.9	38.5 $\pm$ 1.7	32.8 $\pm$ 2.1	42.7
Manual Prompting	55.3 $\pm$ 0.8	62.5 $\pm$ 0.7	48.7 $\pm$ 1.2	38.2 $\pm$ 1.6	43.1 $\pm$ 1.4	36.5 $\pm$ 1.8	47.4
APE [18]	56.8 $\pm$ 1.0	63.9 $\pm$ 0.8	49.3 $\pm$ 1.3	37.5 $\pm$ 2.1	42.8 $\pm$ 1.6	35.9 $\pm$ 2.0	47.7
OPRO [19]	58.1 $\pm$ 0.9	65.2 $\pm$ 0.6	51.6 $\pm$ 1.1	39.8 $\pm$ 1.8	44.7 $\pm$ 1.3	37.2 $\pm$ 1.9	49.4
EvoPrompt [20]	59.4 $\pm$ 0.7	66.1 $\pm$ 0.5	52.9 $\pm$ 1.0	41.3 $\pm$ 1.5	46.2 $\pm$ 1.2	38.6 $\pm$ 1.7	50.8
EvoPrompt-LR (Ours)*	67.8 $\pm$ 0.5†	71.6 $\pm$ 0.4†	63.4 $\pm$ 0.7†	50.4 $\pm$ 1.0†	56.3 $\pm$ 0.9†	49.8 $\pm$ 1.2†	59.9
SFT (LoRA, full data)	70.2 $\pm$ 0.3	74.8 $\pm$ 0.3	61.5 $\pm$ 0.6	48.1 $\pm$ 0.8	54.9 $\pm$ 0.7	45.2 $\pm$ 1.1	59.1

*Best results.

† indicates statistically significant improvement over the best baseline ($p < 0.05$).

Notably, EvoPrompt-LR surpasses SFT with LoRA on Swahili (+1.9%), Yoruba (+2.3%), Uzbek (+1.4%), and Khmer (+4.6%), despite using only 100 labeled examples compared to the 5,000–50,000 examples used for SFT. On Arabic and Vietnamese, where more pretraining data exists, the gap between EvoPrompt-LR and SFT narrows to 2.4% and 3.2%, respectively.

Table 3. NER results (Entity F1 %) LLaMA-3-8B.

Method	Arabic (Dial.)	Vietnamese	Swahili	Yoruba	Uzbek	Khmer	Avg.
Zero-shot	22.4 ± 2.1	31.7 ± 1.8	18.3 ± 2.5	12.1 ± 3.0	14.6 ± 2.8	10.2 ± 3.2	18.2
Few-shot (K=5)	31.8 ± 1.6	39.2 ± 1.4	25.7 ± 2.0	18.4 ± 2.4	21.3 ± 2.2	16.5 ± 2.7	25.5
Manual Prompting	38.5 ± 1.2	44.6 ± 1.0	31.2 ± 1.5	22.7 ± 2.0	26.8 ± 1.8	20.4 ± 2.3	30.7
APE [18]	39.1 ± 1.3	45.3 ± 1.1	30.8 ± 1.7	21.5 ± 2.2	25.9 ± 1.9	19.7 ± 2.4	30.4
OPRO [19]	41.3 ± 1.0	47.8 ± 0.9	33.5 ± 1.4	24.1 ± 1.9	28.4 ± 1.6	22.0 ± 2.1	32.9
EvoPrompt [20]	42.7 ± 0.9	48.5 ± 0.8	35.1 ± 1.2	25.6 ± 1.7	29.7 ± 1.5	23.3 ± 1.9	34.2
EvoPrompt-LR (Ours)	52.6 ± 0.6†	57.3 ± 0.5†	45.8 ± 0.8†	35.2 ± 1.1†	39.4 ± 1.0†	33.7 ± 1.3†	44.0
SFT (LoRA, full data)	56.8 ± 0.4	61.2 ± 0.3	44.3 ± 0.7	33.5 ± 0.9	37.8 ± 0.8	29.1 ± 1.2	43.8

Table 3 shows the NER results. EvoPrompt-LR achieves particularly strong gains on NER, a task where prompt structure (entity type specifications, output format, and demonstration quality) plays a critical role. The improvement over baseline EvoPrompt ranges from 8.8% (Vietnamese) to 10.4% (Khmer). On four of six languages (Swahili, Yoruba, Uzbek, Khmer), EvoPrompt-LR exceeds the SFT upper bound, suggesting that well-optimized prompts can compensate for the model's limited linguistic knowledge more effectively than fine-tuning on noisy low-resource data.

Table 4. QA results (EM/F1 %) LLaMA-3-8B.

Method	Arabic (Dial.)	Vietnamese	Swahili	Yoruba	Uzbek	Khmer	Avg. F1
Zero-shot	18.3/29.5	24.1/36.8	12.7/22.4	7.2/15.3	9.8/18.6	5.9/13.1	22.6
Few-shot (K=5)	25.6/38.2	31.4/44.7	18.9/30.5	11.8/21.6	14.3/25.1	10.2/19.7	30.0
Manual Prompting	30.2/43.8	36.7/50.3	23.4 / 35.9	14.6/25.8	18.1/30.4	13.5/24.2	35.1
APE [18]	31.5/44.6	37.2/51.1	22.8/35.2	13.9/24.7	17.4/29.5	12.8/23.1	34.7
OPRO [19]	33.4/47.1	39.5/53.6	25.8/38.3	16.2/27.9	19.7/32.1	14.6/25.8	37.5
EvoPrompt [20]	34.8/48.5	40.9/55.2	27.3/40.1	17.5/29.4	21.2/33.7	15.8/27.3	39.0
EvoPrompt-LR (Ours)	43.2/58.7†	49.6/64.1†	36.1/50.8†	25.3/39.6†	29.8/43.5†	23.4/37.9†	49.1
SFT (LoRA, full data)	47.5/62.3	53.8/68.4	34.7/48.9	22.8/36.1	27.5/41.3	20.1/33.6	48.4

Table 4 reports the QA results, where EvoPrompt-LR achieves an average F1 improvement of 10.1 percentage points over baseline EvoPrompt. The gains are especially pronounced on Yoruba (+10.2% F1) and Khmer (+10.6% F1), confirming the hypothesis that our framework provides the greatest benefit on languages where LLMs have the least pretraining coverage. Notably, EvoPrompt-LR outperforms SFT on four of six languages for QA.

Table 5. Text summarization results (ROUGE-L/BERTScore F1 %) for LLaMA-3-8B.

Method	Arabic (Dial.)	Vietnamese	Swahili	Yoruba	Uzbek	Khmer	Avg. R-L
Zero-shot	12.8/58.3	16.4/62.1	9.7/54.2	6.3/48.7	7.9/51.4	5.1/46.8	9.7
Few-shot (K=5)	16.5/62.7	20.8/66.4	13.2/58.9	8.7/52.3	10.4/55.1	7.3/50.6	12.8
Manual Prompting	19.4/65.8	23.6/69.2	15.8/61.5	10.5/54.9	12.7/57.8	9.1/53.2	15.2
APE [18]	19.8/66.1	24.1/69.5	15.3/61.0	10.1/54.2	12.2/57.1	8.7/52.5	15.0
OPRO [19]	21.3/67.5	25.7/70.8	17.1/63.2	11.4/56.1	13.5/58.9	10.0/54.3	16.5
EvoPrompt [20]	22.1/68.2	26.9/71.5	18.4/64.1	12.2/57.0	14.3/59.7	10.8/55.1	17.5
EvoPrompt-LR (Ours)	28.5/72.6†	33.2/76.3†	24.7/69.4†	17.8/63.2†	20.1/65.1†	16.3/61.8†	23.4
SFT (LoRA, full data)	30.1/74.2	35.6/78.1	23.5/67.8	15.4/60.5	18.7/63.4	13.8/58.9	22.9

Table 5 presents the summarization results. This task shows the clearest advantage of EvoPrompt-LR's hierarchical genome, as effective summarization prompts require careful coordination between instruction specificity, demonstration selection, CoT scaffolding (for extractive reasoning), and output format constraints. EvoPrompt-LR improves ROUGE-L by 5.8–6.4 points over baseline EvoPrompt across languages and surpasses SFT on all languages except Arabic and Vietnamese.

5.2 | Cross-Model Analysis

Table 6 compares EvoPrompt-LR performance across the three base models, averaged over all four tasks.

Table 6. Average performance (task-macro F1 %) of EvoPrompt-LR across base models, averaged over four tasks.

Model	Arabic (Dial.)	Vietnamese	Swahili	Yoruba	Uzbek	Khmer	Avg.
LLaMA-3-8B	52.1	58.3	46.2	37.6	39.8	35.4	44.9
Mistral-7B	49.8	55.7	43.5	34.1	37.2	32.6	42.2
AceGPT-7B	58.4	47.2	39.8	30.5	34.6	28.9	39.9
LLaMA-3-8B + EvoPrompt-LR	62.3	66.8	56.4	46.9	48.7	44.6	54.3
Mistral-7B + EvoPrompt-LR	59.6	64.1	53.8	43.5	46.1	41.8	51.5
AceGPT-7B + EvoPrompt-LR	68.7	55.9	49.2	39.1	43.3	37.5	48.9

Several observations emerge from Table 6. First, EvoPrompt-LR provides consistent improvements across all three models, with relative gains of 19–25% over the unoptimized baseline. Second, the language-specialized AceGPT-7B achieves the best Arabic performance when combined with EvoPrompt-LR (68.7%), demonstrating that prompt optimization and model specialization are complementary. Third, LLaMA-3-8B benefits most from EvoPrompt-LR on average, likely due to its broader multilingual pretraining enabling better exploitation of cross-lingual fitness transfer. Fourth, the relative improvement from EvoPrompt-LR is inversely correlated with baseline model performance (Pearson $r = -0.73$), confirming that prompt optimization is most valuable when model capability is limited.

5.3 | Ablation Studies

We conduct comprehensive ablation studies to quantify the contribution of each component of EvoPrompt-LR. Table 7 reports the results, averaged across all six languages and four tasks, using LLaMA-3-8B.

Table 7. Ablation study results (average F1 % across six languages and four tasks, LLaMA-3-8B).

Configuration	Avg. F1	Δ
EvoPrompt-LR (Full)	54.3	
Hierarchical genome $\rightarrow$ flat instruction only	46.1	−8.2
Cross-lingual fitness transfer	52.8	−1.5
Semantic diversity preservation	50.9	−3.4
Language-specific mutation operators	49.7	−4.6
SA acceptance (GA only)	51.4	−2.9
Level 2 (demonstration config)	48.5	−5.8
Level 3 (CoT scaffold)	50.2	−4.1
Level 4 (output format)	53.1	−1.2
Random search (same eval budget)	43.8	−10.5

Δ indicates the drop from the full EvoPrompt-LR system.

The ablation results reveal that the hierarchical genome encoding is the single most important component, with its removal causing an 8.2-point drop. Within the genome, the demonstration configuration (*Level 2*) contributes most (−5.8 when removed), followed by the CoT scaffold (*Level 3*, −4.1), confirming that what

examples to show and how to structure reasoning are more impactful than instruction phrasing or output format for low-resource languages. Language-specific mutation operators contribute 4.6 points, validating the importance of linguistically informed search operations. The SA component adds 2.9 points over pure GA, consistent with its role in escaping local optima. Cross-lingual fitness transfer contributes a modest 1.5 points to performance but provides an approximately 5 $\times$ reduction in computational cost (Section 5.4).

5.4 | Convergence and Computational Analysis

Table 8 reports the computational costs and convergence characteristics of EvoPrompt-LR compared to baselines.

Table 8. Computational cost comparison (average across tasks, per language).
GPU-hours on NVIDIA A100.

Method	GPU-Hours	LLM Queries	Low-Resource Evals	Avg. F1
Manual Prompting	0.5	~50	50	37.1
APE	3.2	~800	800	37.0
OPRO	4.8	~1,200	1,200	39.1
EvoPrompt	6.1	~1,500	1,500	40.4
EvoPrompt-LR (w/o transfer)	18.4	~5,000	5,000	52.8
EvoPrompt-LR (Full)	14.6	~5,000	~1,000	54.3
SFT (LoRA)	8.2		Full train set	47.2

EvoPrompt-LR requires 14.6 GPU-hours per language-task pair, which is higher than simpler baselines but provides a substantially better performance-to-cost ratio. The cross-lingual fitness transfer reduces low-resource evaluations from ~5,000 to ~1,000 (an 80% reduction) while actually improving performance by 1.5 points on average, as the proxy evaluation provides a less noisy fitness signal by leveraging larger evaluation sets in the proxy language.

Convergence analysis reveals that EvoPrompt-LR typically reaches 90% of its final performance within 40 generations and 95% within 60 generations. The SA component is most active during generations 20–50, when the temperature is sufficiently low to be selective but still allows occasional acceptance of inferior solutions. The diversity injection mechanism (every 10 generations) produces observable fitness improvements in approximately 30% of injections, indicating that it successfully introduces beneficial genetic material.

6 | Discussion

6.1 | Why Does EvoPrompt-LR Work?

The effectiveness of EvoPrompt-LR can be attributed to several interacting factors. First, the hierarchical genome encoding allows the evolutionary search to explore a structured prompt space rather than the flat, astronomically large space of all possible token sequences. By decomposing prompts into four functionally distinct levels, EvoPrompt-LR enables building-block assembly [37]: good instruction templates can combine with good demonstration configurations and good CoT scaffolds, creating synergistic prompt structures that would be extremely unlikely to arise through random or gradient-free search on a flat representation.

Second, the language-specific mutation operators ensure that the evolutionary search remains within the space of linguistically valid prompts. Our experiments with ablated (language-agnostic) mutation operators revealed that approximately 35% of mutated prompts in Arabic and 28% in Uzbek were grammatically ill-formed, leading to both wasted evaluations and misleading fitness signals. By constraining mutations to respect morphological and syntactic rules, we simultaneously improve search efficiency and fitness estimation quality.

Third, the cross-lingual fitness transfer exploits a non-obvious insight: The structural effectiveness of a prompt (e.g., the number and ordering of demonstrations, the depth of CoT scaffolding, the granularity of

output format specification) is more language-agnostic than the lexical content. This property allows the proxy evaluation to serve as an effective filter even when the proxy and target languages are typologically distant, as confirmed by the positive Spearman correlations (0.62–0.81) observed in our calibration experiments.

6.2 | When Does EvoPrompt-LR Fail?

Despite its overall effectiveness, EvoPrompt-LR exhibits limitations in several scenarios. First, on Arabic and Vietnamese, the two highest-resource languages in our evaluation, the gap between EvoPrompt-LR and SFT remains significant for classification tasks (2–4%), suggesting that when sufficient training data exists, parameter-level adaptation captures linguistic knowledge that cannot be fully expressed through prompts alone.

Second, for tasks requiring extensive world knowledge about the low-resource language's culture or geography (e.g., QA questions about local entities in Khmer or Yoruba), no prompt optimization can compensate for the fundamental absence of this knowledge in the model's parameters. In these cases, EvoPrompt-LR often produces prompts that effectively instruct the model to state "I don't know" rather than hallucinate, which improves precision but reduces recall.

Third, the template grammar approach for instruction generation (*Level 1*) requires manual construction by native-speaking NLP researchers, which itself is a resource-intensive process. While the grammars are reusable across tasks and models, the initial construction cost (estimated at 20–40 person-hours per language) may be prohibitive for languages with very few NLP researchers.

6.3 | Implications for Low-Resource Natural Language Processing

Our results have several important implications for the broader low-resource NLP community. First, the finding that optimized prompts can match or exceed SFT on genuinely low-resource languages (Yoruba, Khmer, Uzbek, Swahili) challenges the prevailing assumption that LLM adaptation necessarily requires labeled data at scale. This finding is particularly significant for the estimated 4,000+ languages classified in Joshi et al.'s [12] lowest resource tiers, where even collecting 100 labeled examples may be feasible but assembling thousands is not.

Second, the effectiveness of cross-lingual fitness transfer suggests a promising research direction in which prompt optimization infrastructure developed for high-resource languages can be systematically repurposed for low-resource targets, reducing the marginal cost of extending NLP capabilities to new languages.

Third, the strong performance of language-specific mutation operators validates the importance of incorporating typological knowledge into NLP systems, even in the era of language-agnostic pretrained models. This finding aligns with recent arguments for typologically informed NLP (Ponti et al., 2019; Bjerva et al., 2019) and suggests that the benefits of typological awareness extend beyond model architecture to the optimization process itself.

6.4 | Limitations

Several limitations of this study should be acknowledged. First, our experiments are limited to models in the 7–8 billion parameter range; the effectiveness of EvoPrompt-LR on larger models (e.g., 70B parameters) or smaller models (e.g., 1–3B parameters) remains to be investigated. Second, the custom datasets for Uzbek and Khmer, while carefully annotated, are relatively small and may not fully represent the linguistic diversity of these languages. Third, our cross-lingual fitness transfer currently uses only English and Chinese as proxy languages; incorporating additional proxy languages (e.g., Russian for Uzbek, Thai for Khmer) could further improve transfer quality. Fourth, the computational cost of EvoPrompt-LR, while reduced by the cross-lingual transfer mechanism, remains substantially higher than simple prompting approaches, which may limit its accessibility for resource-constrained research groups.

7 | Conclusion

We have presented EvoPrompt-LR, a metaheuristic framework for optimizing prompt structures for LLMs in low-resource language settings. Through a hierarchical prompt genome encoding, a hybrid GA-SA optimization algorithm, cross-lingual fitness transfer, language-specific mutation operators, and semantic diversity preservation, EvoPrompt-LR achieves substantial improvements over existing prompt optimization methods across six low-resource languages and four NLP tasks. On the most resource-constrained languages (Yoruba, Khmer), improvements of 19–22% over the best baseline demonstrate the potential of intelligent prompt optimization as a practical and cost-effective approach to extending LLM capabilities to underrepresented languages.

Our findings suggest that the structure of prompts how instructions are phrased, which examples are shown and in what order, how reasoning is scaffolded, and what output format is specified constitutes a rich and underexplored optimization space that is particularly impactful in low-resource settings. By bringing together evolutionary computation, cross-lingual transfer, and typological linguistics, EvoPrompt-LR provides a principled framework for navigating this space.

Future work will extend EvoPrompt-LR in several directions: 1) Scaling to larger LLMs (70B+ parameters) and investigating whether prompt optimization and model scaling exhibit complementary or diminishing returns, 2) Incorporating human-in-the-loop feedback to guide the evolutionary search based on qualitative assessments of prompt naturalness and cultural appropriateness, 3) Extending the framework to multilingual and code-switching scenarios, where prompts may need to combine elements from multiple languages, and 4) developing automated methods for template grammar construction to reduce the manual effort required for new languages.

Authors' Contributions

All aspects of the research and manuscript preparation were carried out by the author. The author has read and approved the final version of the manuscript.

Funding

Not applicable.

Data Availability

All data supporting the reported findings in this research paper are provided within the manuscript.

Conflict of Interest

The author declares that they do not have any conflict of interest.

Consent for Publication

The author confirms consent for the publication of this work.

Ethics Approval and Consent to Participate

This article does not contain any studies with human participants performed by the author.

References

- [1] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... , & Amodei, D. (2020). Language models are few-shot learners. *Advances in neural information processing systems* (Vol. 33, pp. 1877–1901). NeurIPS Foundation

- https://proceedings.neurips.cc/paper_files/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html
- [2] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M. A., Lacroix, T., ... , & Lample, G. (2023). *LLaMA: Open and efficient foundation language models*. <https://doi.org/10.48550/arXiv.2302.13971>
 - [3] Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., ... , & Scialom, T. (2023). *Llama 2: Open foundation and fine-tuned chat models*. <https://doi.org/10.48550/arXiv.2307.09288>
 - [4] Meta AI. (2024). *Introducing meta Llama 3: The most capable openly available LLM to date*. <https://ai.meta.com/blog/meta-llama-3/>
 - [5] Jiang, Y., Li, X., Zhu, G., Li, H., Deng, J., Han, K., ... , & Zhang, R. (2023). *Mistral 7B*. <https://doi.org/10.48550/arXiv.2310.06825>
 - [6] Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., & Neubig, G. (2023). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM computing surveys*, 55(9), 1–35. <https://doi.org/10.1145/3560815>
 - [7] Zhao, Z., Wallace, E., Feng, S., Klein, D., & Singh, S. (2021). Calibrate before use: Improving few-shot performance of language models. *Proceedings of the 38th international conference on machine learning* (Vol. 139, pp. 12697–12706). PMLR. <https://proceedings.mlr.press/v139/zhao21c.html>
 - [8] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., ... , & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems* (Vol. 35, pp. 24824–24837). Curran Associates, Inc. <https://doi.org/10.52202/068431-1800>
 - [9] Wang, X., Wei, J., Schuurmans, D., Le, Q. V., Chi, E. H., Narang, S., Chowdhery, A., & Zhou, D. (2023). Self-consistency improves chain of thought reasoning in language models. *The eleventh international conference on learning representations (ICLR 2023)*. OpenReview. <https://openreview.net/forum?id=1PL1NIMMrw>
 - [10] Yao, S., Yu, D., Zhao, J., Shafran, I., Griffiths, T., Cao, Y., & Narasimhan, K. (2023). Tree of thoughts: Deliberate problem solving with large language models. *Advances in neural information processing systems* (Vol. 36, pp. 11809–11822). Curran Associates, Inc. <https://doi.org/10.52202/075280-0517>
 - [11] Zhang, Z., Zhang, A., Li, M., & Smola, A. (2023). Automatic chain of thought prompting in large language models. *The eleventh international conference on learning representations (ICLR 2023)*. OpenReview. <https://openreview.net/forum?id=5NTt8GFjUHkr>
 - [12] Joshi, P., Santy, S., Budhiraja, A., Bali, K., & Choudhury, M. (2020). The state and fate of linguistic diversity and inclusion in the NLP world. *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 6282–6293). Online: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2020.acl-main.560>
 - [13] Ahuja, K., Diddee, H., Hada, R., Ochieng, M., Ramesh, K., Jain, P., ... , & Sitaram, S. (2023). MEGA: Multilingual evaluation of generative AI. *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 4232–4267). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.emnlp-main.258>
 - [14] Lai, V. D., Ngo, N., Pouran Ben Veyseh, A., Man, H., Démoncourt, F., Bui, T., & Nguyen, T. H. (2023). ChatGPT beyond English: Towards a comprehensive evaluation of large language models in multilingual learning. *Findings of the association for computational linguistics: EMNLP 2023* (pp. 13171–13189). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.findings-emnlp.878>
 - [15] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., ... , & Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 8440–8451). Online: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2020.acl-main.747>
 - [16] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. *Proceedings of the 2019 conference of the north American chapter of the association for computational linguistics: Human language technologies, Volume 1 (long and short papers)* (pp. 4171–4186). Minneapolis, Minnesota: Association for Computational Linguistics. <https://doi.org/10.18653/v1/N19-1423>

- [17] Pfeiffer, J., Goyal, N., Lin, X. V., Li, X., Cross, J., Riedel, S., & Artetxe, M. (2022). Lifting the curse of multilinguality by pre-training modular transformers. *Proceedings of the 2022 conference of the north american chapter of the association for computational linguistics: human language technologies* (pp. 3479–3495). Seattle, United States: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2022.naacl-main.255>
- [18] Zhou, Y., Muresanu, A. I., Han, Z., Paster, K., Pitis, S., Chan, H., & Ba, J. (2023). Large language models are human-level prompt engineers. *The eleventh international conference on learning representations (ICLR 2023)*. OpenReview. <https://openreview.net/forum?id=92gvk82DE->
- [19] Yang, C., Wang, X., Lu, Y., Liu, H., Le, Q. V., Zhou, D., & Chen, X. (2024). Large language models as optimizers. *International conference on learning representations* (Vol. 2024, pp. 12028–12068). OpenReview.net. https://proceedings.iclr.cc/paper_files/paper/2024/file/3339f19c5fcee3ad74502947a32be9e6-Paper-Conference.pdf
- [20] Guo, Q., Wang, R., Guo, J., Li, B., Song, K., Tan, X., ... , & Yang, Y. (2024). Connecting large language models with evolutionary algorithms yields powerful prompt optimizers. *International conference on learning representations* (Vol. 2024, pp. 34133–34156). OpenReview.net. https://proceedings.iclr.cc/paper_files/paper/2024/file/9156b0f6dfa9bbd18c79cc459ef5d61c-Paper-Conference.pdf
- [21] Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002). A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE transactions on evolutionary computation*, 6(2), 182–197. <https://doi.org/10.1109/4235.996017>
- [22] Lu, Y., Bartolo, M., Moore, A., Riedel, S., & Stenetorp, P. (2022). Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *Proceedings of the 60th annual meeting of the association for computational linguistics (Volume 1: Long papers)* (pp. 8086–8098). Dublin, Ireland: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2022.acl-long.556>
- [23] Li, X. L., & Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th annual meeting of the association for computational linguistics and the 11th international joint conference on natural language processing (Volume 1: Long papers)* (pp. 4582–4597). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.acl-long.353>
- [24] Lester, B., Al-Rfou, R., & Constant, N. (2021). The power of scale for parameter-efficient prompt tuning. *Proceedings of the 2021 conference on empirical methods in natural language processing* (pp. 3045–3059). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.emnlp-main.243>
- [25] Pryzant, R., Iter, D., Li, J., Lee, Y., Zhu, C., & Zeng, M. (2023). Automatic prompt optimization with "gradient descent" and beam search. *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 7957–7968). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.emnlp-main.494>
- [26] Fernando, C., Banarse, D. S., Michalewski, H., Osindero, S., & Rocktäschel, T. (2024). Promptbreeder: Self-referential self-improvement via prompt evolution. *Proceedings of the 41st international conference on machine learning* (Vol. 235, pp. 13481–13544). PMLR. <https://proceedings.mlr.press/v235/fernando24a.html>
- [27] Feng, S. Y., Gangal, V., Wei, J., Chandar, S., Vosoughi, S., Mitamura, T., & Hovy, E. (2021). A survey of data augmentation approaches for NLP. *Findings of the association for computational linguistics: ACL-IJCNLP 2021* (pp. 968–988). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.findings-acl.84>
- [28] Pires, T., Schlinger, E., & Garrette, D. (2019). How multilingual is multilingual BERT? *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4996–5001). Florence, Italy: Association for Computational Linguistics. <https://doi.org/10.18653/v1/P19-1493>
- [29] Wu, S., & Dredze, M. (2019). Beto, Bentz, Becas: The surprising cross-lingual effectiveness of BERT. *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing (EMNLP-IJCNLP)* (pp. 833–844). Hong Kong, China: Association for Computational Linguistics. <https://doi.org/10.18653/v1/D19-1077>

- [30] Xue, L., Constant, N., Roberts, A., Kale, M., Al-Rfou, R., Siddhant, A., ... , & Raffel, C. (2021). mT5: A massively multilingual pre-trained text-to-text transformer. *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics: Human language technologies* (pp. 483–498). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.naacl-main.41>
- [31] OpenAI, A. (2023). *GPT-4 technical report*. <https://doi.org/10.48550/arXiv.2303.08774>
- [32] BigScience Workshop. (2022). *Bloom: A 176b-parameter open-access multilingual language model*. <https://doi.org/10.48550/arXiv.2211.05100>
- [33] Zhu, W., Liu, H., Dong, Q., Xu, J., Huang, S., Kong, L., ... , & Li, L. (2024). Multilingual machine translation with large language models: Empirical results and analysis. *Findings of the association for computational linguistics: NAACL 2024* (pp. 2765–2781). Mexico City, Mexico: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2024.findings-naacl.176>
- [34] Huang, H., Yu, F., Zhu, J., Sun, X., Cheng, H., Dingjie, S., ... , & Xu, J. (2024). AceGPT, localizing large language models in Arabic. *Proceedings of the 2024 conference of the north American chapter of the association for computational linguistics: Human language technologies (Volume 1: Long papers)* (pp. 8139–8163). Mexico City, Mexico: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2024.naacl-long.450>
- [35] AI Singapore. (2024). *SEA-LION: Southeast Asian languages in one network*. <https://github.com/aisingapore/sealion>
- [36] Dossou, B. F. P., Tonja, A. L., Yousuf, O., Osei, S., Oppong, A., Shode, I., Awoyomi, O. O., & Emezue, C. C. (2022). AfroLM: A self-active learning-based multilingual pretrained language model for 23 African languages. *Proceedings of the third workshop on simple and efficient natural language processing (SustainLP)* (pp. 52–64). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2022.sustainlp-1.11>
- [37] Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Addison-Wesley Longman Publishing. <https://dl.acm.org/doi/book/10.5555/534133>
- [38] Ghareb, A. S., Bakar, A. A., & Hamdan, A. R. (2016). Hybrid feature selection based on enhanced genetic algorithm for text categorization. *Expert systems with applications*, 49, 31–47. <https://doi.org/10.1016/j.eswa.2015.12.004>
- [39] Smith, T. C., & Witten, I. H. (1995). A genetic algorithm for the induction of natural language grammars. *Proceedings of the IJCAI-95 workshop on new approaches to learning for natural language processing* (pp. 17–24). Montreal, Canada.
- [40] Christy, A., & Thambidurai, P. (2006). Efficient information extraction using machine learning and classification using genetic and C4. 8 algorithms. *Information technology journal*, 5, 1023–1027. <https://doi.org/10.3923/itj.2006.1023.1027>
- [41] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671–680. <https://doi.org/10.1126/science.220.4598.671>
- [42] Cowie, J., Guthrie, J., & Guthrie, L. (1992). Lexical disambiguation using simulated annealing. *COLING 1992: Proceedings of the 14th international conference on computational linguistics* (Vol. 1, pp. 359–365). Association for Computational Linguistics. <https://doi.org/10.3115/992066.992125>
- [43] Aliguliyev, R. M. (2009). A new sentence similarity measure and sentence based extractive technique for automatic text summarization. *Expert systems with applications*, 36(4), 7764–7772. <https://doi.org/10.1016/j.eswa.2008.11.022>
- [44] Hermann, U., Jahr, M., Knight, K., Marcu, D., and Yamada, K. (2001). Fast decoding and optimal decoding for machine translation. *Proceedings of the 39th annual meeting of the association for computational linguistics (ACL)* (pp. 228–235). Association for Computational Linguistics. <https://doi.org/10.3115/1073012.1073042>
- [45] Mahfoud, S. W., & Goldberg, D. E. (1995). Parallel recombinative simulated annealing: A genetic algorithm. *Parallel computing*, 21(1), 1–28. [https://doi.org/10.1016/0167-8191\(94\)00071-H](https://doi.org/10.1016/0167-8191(94)00071-H)
- [46] Li, W. D., Ong, S. K., & Nee, A. Y. C. (2002). Hybrid genetic algorithm and simulated annealing approach for the optimization of process plans for prismatic parts. *International journal of production research*, 40(8), 1899–1922. <https://doi.org/10.1080/00207540110119991>
- [47] Ruder, S., Vulić, I., & Søgaard, A. (2019). A survey of cross-lingual word embedding models. *Journal of artificial intelligence research*, 65, 569–631. <https://doi.org/10.1613/jair.1.11640>

- [48] Pfeiffer, J., Vulić, I., Gurevych, I., & Ruder, S. (2020). MAD-X: An adapter-based framework for multi-task cross-lingual transfer. *Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP)* (pp. 7654–7673). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2020.emnlp-main.617>
- [49] Huang, H., Tang, T., Zhang, D., Zhao, X., Song, T., Xia, Y., & Wei, F. (2023). Not all languages are created equal in LLMs: Improving multilingual capability by cross-lingual-thought prompting. *Findings of the association for computational linguistics: EMNLP 2023* (pp. 12365–12394). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.findings-emnlp.826>
- [50] Qin, L., Chen, Q., Wei, F., Huang, S., & Che, W. (2023). Cross-lingual prompting: improving zero-shot chain-of-thought reasoning across languages. *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 2695–2709). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.emnlp-main.163>
- [51] Shi, F., Suzgun, M., Freitag, M., Wang, X., Srivats, S., Vosoughi, S., Chung, H. W., Tay, Y., Ruder, S., Zhou, D., Das, D., & Wei, J. (2023). Language models are multilingual chain-of-thought reasoners. *The eleventh international conference on learning representations (ICLR 2023)*. OpenReview.net. <https://dblp.org/rec/conf/iclr/ShiSF0SVCTRZ0W23.html>
- [52] Winata, G. I., Huang, L., & Vadamannati, S Chandarana, Y. (2023). *Multilingual few-shot learning via language model retrieval*. <https://doi.org/10.48550/arXiv.2306.10964>
- [53] Davis, L. (1985). Applying adaptive algorithms to epistatic domains. *IJCAI* (Vol. 1, pp. 162–164). ACM Digital Library. <https://doi.org/10.5555/1625135.1625164>
- [54] Costa-Jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heafield, K., Heffernan, K., ... , & Wang, J. (2022). *No language left behind: Scaling human-centered machine translation*. <https://doi.org/10.48550/arXiv.2207.04672>
- [55] Batsuren, K., Goldman, O., Khalifa, S., Habash, N., Kieraś, W., Bella, G., ... Vylomova, E. (2022). UniMorph 4.0: Universal morphology. *Proceedings of the thirteenth language resources and evaluation conference* (pp. 840–855). Marseille, France: European Language Resources Association. <https://aclanthology.org/2022.lrec-1.89/>
- [56] Feng, F., Yang, Y., Cer, D., Arivazhagan, N., & Wang, W. (2022). Language-agnostic bert sentence embedding. *Proceedings of the 60th annual meeting of the association for computational linguistics (Volume 1: Long papers)* (pp. 878–891). Dublin, Ireland: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2022.acl-long.62>
- [57] Črepinšek, M., Liu, S.-H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM computing surveys (CSUR)*, 45(3), 1–33. <https://doi.org/10.1145/2480741.2480752>
- [58] Baly, R., Khaddaj, A., Hajj, H., El-Hajj, W., & Shaban, K. B. (2018). *ArSentD-LEV: A multi-topic corpus for target-based sentiment analysis in Arabic levantine tweets*. <https://doi.org/10.48550/arXiv.1906.01830>
- [59] Benajiba, Y., Rosso, P., & BenedíRuiz, J. M. (2007). ANERsys: An Arabic named entity recognition system based on maximum entropy. *Computational linguistics and intelligent text processing* (pp. 143–153). Berlin, Heidelberg: Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-540-70939-8_13
- [60] Clark, J. H., Choi, E., Collins, M., Garrette, D., Kwiatkowski, T., Nikolaev, V., & Palomaki, J. (2020). TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the association for computational linguistics*, 8, 454–470. https://doi.org/10.1162/tacl_a_00317
- [61] Hasan, T., Bhattacharjee, A., Islam, M. S., Mubasshir, K., Li, Y. F., Kang, Y. B., ... , & Shahriyar, R. (2021). XL-sum: Large-scale multilingual abstractive summarization for 44 languages. *Findings of the association for computational linguistics: ACL-IJCNLP 2021* (pp. 4693–4703). Association for Computational Linguistics (ACL). <https://doi.org/10.18653/v1/2021.findings-acl.413>
- [62] Nguyen, K. Van, Nguyen, V. D., Nguyen, P. X. V., Truong, T. T. H., & Nguyen, N. L. T. (2018). UIT-VSFC: Vietnamese students' feedback corpus for sentiment analysis. *2018 10th international conference on knowledge and systems engineering (KSE)* (pp. 19–24). IEEE. <https://doi.org/10.1109/KSE.2018.8573337>
- [63] Truong, T. H., Dao, M. H., & Nguyen, D. Q. (2021). COVID-19 named entity recognition for vietnamese. *Proceedings of the 2021 conference of the north american chapter of the association for computational linguistics:*

- Human language technologies* (pp. 2146–2153). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.naacl-main.173>
- [64] Van Nguyen, K., Nguyen, D. V., Gia-Tuan Nguyen, A., & Luu-Thuy Nguyen, N. (2020). A vietnamese dataset for evaluating machine reading comprehension. *Proceedings of the 28th international conference on computational linguistics* (pp. 2595–2605). Barcelona, Spain: International Committee on Computational Linguistics. <https://doi.org/10.18653/v1/2020.coling-main.233>
- [65] Muhammad, S. H., Abdulmumin, I., Ayele, A. A., Ousidhoum, N., Adelani, D. I., Yimam, S. M., ... , & Arthur, S. (2023). AfriSenti: A twitter sentiment analysis benchmark for african languages. *Proceedings of the 2023 conference on empirical methods in natural language processing* (pp. 13968–13981). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.emnlp-main.862>
- [66] Adelani, D. I., Abbott, J., Neubig, G., D'souza, D., Kreutzer, J., Lignos, C., ... , & Osei, S. (2021). MasakhaNER: Named entity recognition for African languages. *Transactions of the association for computational linguistics*, 9, 1116–1131. https://doi.org/10.1162/tacl_a_00416
- [67] Ogundepo, O., Gwadabe, T. R., Rivera, C. E., Clark, J. H., Ruder, S., Adelani, D. I., ... , & Adhiambo, S. (2023). AfriQA: Cross-lingual open-retrieval question answering for african languages. *The 2023 conference on empirical methods in natural language processing (EMNLP 2023)* (pp. 14957–14972). Singapore: Association for Computational Linguistics. <https://doi.org/10.18653/v1/2023.findings-emnlp.997>
- [68] Kuriyozov, E., & Matlatipov, S. (2019). Building a New Sentiment Analysis Dataset for Uzbek Language and Creating Baseline Models. *Proceedings*, 21(1), 37. <https://doi.org/10.3390/proceedings2019021037>
- [69] White, J., Fu, Q., Hays, S., Sandborn, M., Olea, C., Gilbert, H., ... , & Schmidt, D. C. (2023). *A prompt pattern catalog to enhance prompt engineering with chatgpt*. <https://doi.org/10.48550/arXiv.2302.11382>
- [70] Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., & Chen, W. (2022). LoRA: Low-rank adaptation of large language models. *Proceedings of the tenth international conference on learning representations (ICLR 2022)*. OpenReview.net. <https://openreview.net/pdf?id=nZeVKeeFYf9>
- [71] Lin, C. Y. (2004). ROUGE: A package for automatic evaluation of summaries. *Text summarization branches out*, 74–81. Association for Computational Linguistics. <https://aclanthology.org/W04-1013/>
- [72] Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2020). BERTScore: Evaluating text generation with BERT. *Proceedings of the eighth international conference on learning representations (ICLR 2020)*. OpenReview.net. <https://openreview.net/pdf/15e539377cdf8d827f6e08cb7de3f36a618d824b.pdf>
- [73] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., ... , & Rush, A. (2020). Transformers: State-of-the-art natural language processing. *Proceedings of the 2020 conference on empirical methods in natural language processing: System demonstrations* (pp. 38–45). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2020.emnlp-demos.6>