Prompt Engineering for Nepali NER: Leveraging Hindi-Capable LLMs for Low-Resource Languages

Dipendra Yadav[1], **Sumaiya Suravee**[1], **Stefan Kemnitz**[1],
Tobias Strauß[2], **Kristina Yordanova**[1]
[1]University of Greifswald, Germany
[2]University of Rostock, Germany
{yadavd, suravees, stefan.kemnitz, kristina.yordanova}@uni-greifswald.de
tobias.strauss@uni-rostock.de

Abstract

This study provides a systematic evaluation of prompt engineering strategies for Named Entity Recognition in Nepali, a low-resource language with high similarity to Hindi, by leveraging Hindi-capable Meta's LLaMA 3.3:70B model. Four prompting techniques—Baseline, Chain-of-Thought, Self-Refine, and Least-to-Most—are assessed in both zero-shot and few-shot settings. As a novel contribution, we propose an entity-aware sentence selection strategy that prioritizes example diversity and entity coverage for few-shot prompting. Experimental results show that, without Nepali examples, zero-shot and one-shot prompts frequently yield unstructured or hallucinated outputs, underscoring the limitations of cross-lingual capabilities without in-context supervision. However, including even a small number of carefully selected Nepali examples—sometimes as few as ten—substantially enhances model performance, with the Least-to-Most approach achieving the highest F1 scores. These findings highlight the potential of prompt-based adaptation and principled example curation for extending LLM capabilities to related, low-resource languages, offering a practical alternative to full model fine-tuning.

1 Introduction

Named Entity Recognition (NER) is a foundational task in natural language processing (NLP), focused on identifying and categorizing named entities such as persons, locations, and organizations in unstructured text (Li et al., 2020). While traditional machine learning and deep learning approaches have achieved robust results for high-resource languages—benefiting from the availability of large annotated corpora—progress in low-resource languages such as Nepali, Maithili, and Bhojpuri remains limited by scarce labelled data and the high annotation costs required for state-of-the-art model training (Singh et al., 2019; Mundotiya et al., 2020).

Recent advances in Large Language Models (LLMs) (Brown, 2020; Smith et al., 2022) and prompt engineering techniques (Li and Liang, 2021; Schick and Schütze, 2020) have introduced promising avenues for addressing these challenges. Prompt-based approaches exploit the representational power of LLMs to perform NER with minimal supervision, typically using carefully structured prompts and, in few-shot scenarios, a small set of annotated examples (Liu et al., 2021; Vilar et al., 2022). Unlike adapter-based or fine-tuning methods, prompt engineering allows direct interaction with frozen LLMs, removing the need for resource-intensive retraining—an appealing feature for low-resource language research where both annotated data and computational resources are often limited.

This study is guided by three main questions: (1) To what extent can a Hindi-capable large language model, without explicit Nepali pretraining, be prompted to perform effective NER in Nepali given their strong linguistic similarity (Beaufils, 2015–2025)? (2) How do different prompt engineering strategies—Baseline (Anthropic, 2025), Chain-of-Thought (Wei et al., 2022), Self-Refine (Madaan et al., 2023), and Least-to-Most (Zhou et al., 2023)—compare under zero-shot and few-shot settings for Nepali NER? (3) Does a principled, entity-aware sentence selection strategy for few-shot examples lead to measurable gains over random sampling in model performance?

To our knowledge, this work presents the first systematic evaluation of prompt engineering techniques for NER in Nepali using a Hindi-capable LLM, and introduces a novel, entity-aware sentence selection strategy for optimizing few-shot prompts. These contributions provide new empirical insights into the feasibility and limitations of

Proceedings of Recent Advances in Natural Language Processing,pages 1364–1373
Varna, Sep 8–10, 2025
https://doi.org/10.26615/978-954-452-098-4-158

prompt-based cross-lingual NER in closely related, low-resource languages.

2 Related Work

Named Entity Recognition (NER) for low-resource languages continues to present substantial challenges, primarily due to the lack of annotated corpora and limited language resources (Murthy et al., 2018). Researchers have attempted to bridge this gap by employing a range of techniques, including transfer learning, data augmentation, multilingual models, and cross-lingual embeddings (Kamath and Vajjala, 2025; Farahani et al., 2021; Qin et al., 2024; Ruder et al., 2019). For instance, Feng et al. (2018) demonstrated that the use of bilingual lexicons and neural architectures can enhance name tagging in languages such as Spanish and Dutch by leveraging English as a source language. Comparable patterns have also emerged within the context of South Asian languages, as evidenced by recent studies (Bhargava et al., 2023; Yadav et al., 2024).

With respect to Nepali, initial efforts were centred around traditional machine learning methods, including Support Vector Machines (SVMs), Naïve Bayes, and bi-directional LSTM architectures (Bam and Shahi, 2014; Maharjan et al., 2019; Singh et al., 2019). The advent of transformer-based architectures and multilingual language models has resulted in marked improvements (Thapa et al., 2025), exemplified by the DanfeNER project (Niraula and Chapagain, 2023), which introduced a benchmark for Nepali NER using annotated tweet datasets. Although recent work has advanced Nepali NER (Subedi et al., 2024), ongoing progress remains limited by the persistent scarcity of labelled data and the morphological complexity inherent to the language.

Owing to the strong linguistic affinity between Hindi and Nepali, recent research has explored cross-lingual transfer learning as a means of mitigating data limitations in Nepali (Yadav et al., 2024). However, these approaches still rely on explicitly training and fine-tuning models for the specific target task. As LLMs continue to grow in scale, the resources required for comprehensive training remain accessible only to a select few with access to significant computational and data resources.

Prompt engineering has recently emerged as a compelling alternative to model fine-tuning, enabling large language models to undertake downstream tasks by means of well-crafted instructions and a small number of examples (Li and Liang, 2021; Ma et al., 2021; Cheng et al., 2024). This method is particularly advantageous in few-shot scenarios for low-resource languages, where full-scale model training is often infeasible (Zhang et al., 2025). Emerging evidence suggests that the quality, structure, and diversity of prompt examples play a critical role in determining model performance in NER tasks (Naguib et al., 2024).

The present study builds on this line of inquiry by examining whether a Hindi-capable LLM—without explicit exposure to Nepali during pretraining—can be effectively prompted to perform NER in Nepali. We systematically compare four prompt engineering strategies, evaluating their efficacy under both zero-shot and few-shot conditions, with particular emphasis on the role of careful example selection. This work provides new empirical insights at the intersection of prompt engineering and cross-lingual NER, shedding light on both the limitations and the potential of LLMs for tackling low-resource language challenges.

3 Methodology

The overall methodology is depicted in Figure 1. The process begins with dataset preparation and splitting into few-shot examples and test sets. The sentence selection algorithm is then applied to identify optimal, entity-rich examples for few-shot prompts. Depending on the experimental setting—zero-shot or few-shot—prompts are constructed accordingly.

For zero-shot prompting, only the test sentence and explicit task instructions are provided, with no annotated examples. In the few-shot scenario, a small set of annotated Nepali examples, selected as described below, is included before the test instance to guide prediction.

Following prompt construction, inputs are submitted to the LLM, which generates a response for each test sentence. A validation step automatically checks if the output adheres to the expected XML format and tag constraints. If the response is invalid, a retry mechanism is triggered: the same prompt is resubmitted up to three times, as the model can produce valid outputs in subsequent attempts due to its autoregressive nature. This mechanism is particularly necessary, as the model sometimes fails to output the strict XML required for automated evaluation. We are not

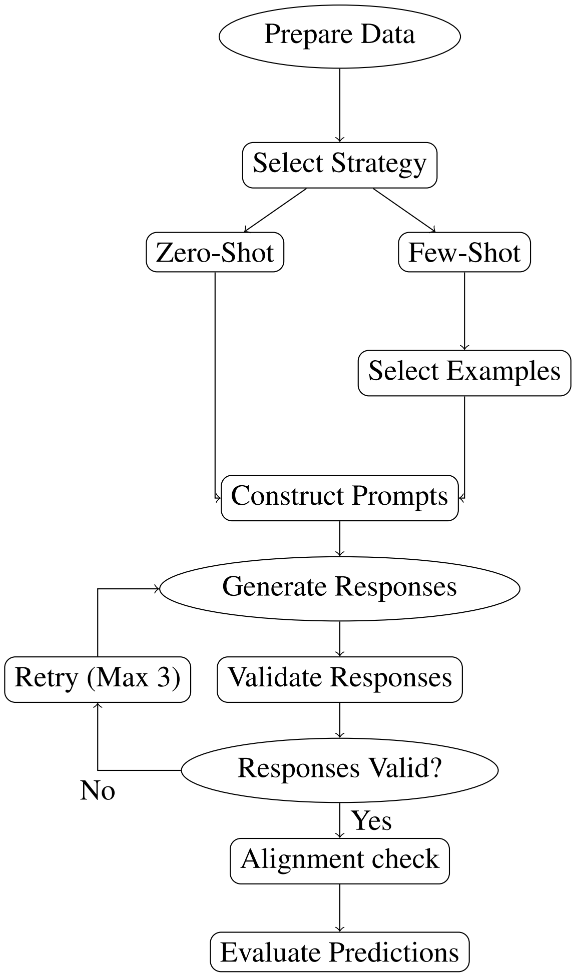


Figure 1: The proposed methodology for exploring prompt engineering.

aware of prior work employing such a retry protocol for output validation in prompt-based NER. Once responses passed validation, alignment checking was performed to verify that the words in the LLM's response matched the words in the test sentences. This step was essential because the model would occasionally omit or modify tokens. Finally, the model's predictions were evaluated against the ground truth annotations using precision, recall, and F1-score metrics to measure performance.

3.1 Sentence Selection Strategy

Our sentence selection strategy, described in Algorithm 1, ensures that few-shot prompts contain diverse and entity-rich examples, covering all named entity types and maximizing tag diversity. This approach is both computationally efficient and deterministic; there is no inherent randomness in selection. It simply sorts and filters sentences based on entity type coverage and tag density. By construction, this strategy increases the representativeness of prompt examples and, as our results demonstrate, consistently outperforms random sampling.

Algorithm 1 Sentence Selection Strategy

```
Input: Set of sentences $S$, required tags $T$, target number $N$
Output: Top $N$ sentences prioritized by completeness and diversity
Group sentences into two sets:
    • $S_{\text{all}}$: Containing all required tags $T$
    • $S_{\text{div}}$: Missing some required tags
Rank sentences in $S_{\text{all}}$ by entity count (descending)
For sentences in $S_{\text{div}}$:
    • Calculate tag diversity as the unique tags per sentence
    • Rank by tag diversity and total number of tags (descending)
$S_{\text{selected}} \leftarrow$ Top $N$ sentences from $S_{\text{all}}$
if $|S_{\text{selected}}| < N$ then
    Add top-ranked sentences from $S_{\text{div}}$ until $|S_{\text{selected}}| = N$
end if
return $S_{\text{selected}}$
```

3.2 Prompt Engineering Techniques

To systematically investigate prompt engineering for Nepali Named Entity Recognition (NER), we implemented four distinct strategies: Baseline Prompting (Anthropic, 2025), Chain-of-Thought (CoT) prompting (Wei et al., 2022), Self-Refine prompting (Madaan et al., 2023), and Least-to-Most prompting (Zhou et al., 2023). Each strategy was evaluated in both zero-shot and few-shot configurations (Sivarajkumar et al., 2024).

The **Baseline Prompting** technique follows a straightforward approach, providing the model with direct task instructions and requiring output in a strictly specified structured format. This method serves as a reference for conventional prompt design, where the model must assign entity tags based solely on the prompt and any prior knowledge.

Chain-of-Thought (CoT) Prompting (Wei et al., 2022) builds on this structure by requiring the model to generate step-by-step natural language reasoning before producing the final output. Here, the model explains how it detects entity boundaries, assigns tags, and applies BIO conventions, before outputting its predictions in the expected structured format.

Self-Refine Prompting (Madaan et al., 2023) adds an iterative, self-correcting layer to the annotation process. The model first generates a draft response, then critiques its own output, and finally produces a repaired version that addresses any detected errors or inconsistencies. This approach encourages more robust annotation by mimicking human revision practices.

Least-to-Most Prompting (Zhou et al., 2023) decomposes the NER task into a series of simpler subproblems. The model first identifies entity spans, then assigns entity types, and finally translates these into token-level BIO tags. This structured, hierarchical approach allows the model to generalize from simple examples to more complex cases and is particularly suited to compositional reasoning.

Baseline Few-Shot Prompt Template

```
<context>You are an expert in identifying named entities in
Nepali text. Below are training examples, followed by a
test sentence.
</context>
<training_examples><ner_tagged_sentence>
  <pair><word>{WORD1}</word><pred_tag>TAG1</pred_tag></pair>
  <pair><word>{WORD2}</word><pred_tag>TAG2</pred_tag></pair>
  ...
</ner_tagged_sentence>
...
</training_examples>
<description>
Each word is tagged as one of: B-LOC, I-LOC, B-ORG, I-ORG,
      B-PER, I-PER, or O.
- 'B-' marks the start of an entity.
- 'I-' marks the continuation of the same entity.
- Every 'I-' tag must follow a matching 'B-' tag (e.g.,
      'I-ORG' after 'B-ORG').
- Correct any case where an 'I-' appears without its
      preceding 'B-'.
</description>
<task>
  <test_sentence>{NEPALI SENTENCE}</test_sentence>
  Analyze the sentence and ensure each word is tagged
  correctly. Confirm that each 'I-' tag has a preceding
  'B-' tag.
</task>
<output_formatting>
The model must respond with **exactly** the XML shown
      below-no extra text!
<tagged_output>
  <pair><word>WORD</word><pred_tag>TAG</pred_tag></pair>
  ...
</tagged_output>
</output_formatting>
```

Figure 2: Prompt template for **Few-Shot + Baseline** used in the experiment. The model is provided with annotated examples and is required to provide only the final structured prediction. The zero-shot version is structurally identical, excluding training examples.

Figures 2–5 present the few-shot prompt templates for each technique. The Baseline prompt defines the fundamental structure shared by all strategies, except for the reasoning section. Zero-shot

Chain-of-Thought Prompt (Few-Shot Excerpt)

```
<task>
<test_sentence>
{NEPALI SENTENCE}
</test_sentence>
Think step by step and place your reasoning inside a
      <reasoning> tag, then output
the final BIO tags inside <tagged_output>.
- Review training examples to internalize tagging patterns.
- Tag each token of the test sentence while explaining your
      decisions.
- Ensure I-tag continuity.
After completing your reasoning, output the final tag list.
</task>
<output_formatting>
The model must respond with exactly two root-level XML
      sections in the following
order:
1. <reasoning> - A detailed, step-by-step explanation of
      how the tag for each word
was decided.
2. <tagged_output> - One <pair> per word, keeping the
      original token order, using:
   <pair><word>WORD</word><pred_tag>TAG</pred_tag></pair>
No text of any kind is allowed outside these two tags.
</output_formatting>
```

Figure 3: Prompt template for **Few-Shot + Chain-of-Thought** showing only parts different from the Baseline template. The model is required to produce stepwise reasoning before the final structured prediction. Zero-shot is structurally identical, minus training examples.

Least-to-Most Prompt (Few-Shot Excerpt)

```
<task>
<test_sentence>
{NEPALI SENTENCE}
</test_sentence>
Inside <reasoning>, solve the task in three explicit steps:
  1. <step1_spans> Detect contiguous entity spans.
     - Output each span as
           <span>start_index-end_index</span>.
  2. <step2_types> Assign entity type PER / ORG / LOC to
       every span.
     - Format:
     <typed_span>
     <span_id>k</span_id><entity_type>PER</entity_type>
     </typed_span>
  3. <step3_bio> Convert the span+type list into
       token-level BIO tags.

After completing all three steps *inside <reasoning>*,
      output only
the final BIO labels inside <tagged_output>, one <pair> per
      token.
</task>
<output_formatting>
The model must respond with exactly two root-level XML
      sections, in order:
1. <reasoning>  any intermediate thinking / sub-steps.
2. <tagged_output>  one <pair> per token:
   <pair><word>WORD</word><pred_tag>TAG</pred_tag></pair>
No text of any kind is allowed outside these two tags.
</output_formatting>
```

Figure 4: Prompt template for **Few-Shot + Least-to-Most** showing only parts different from Baseline. The model is guided through explicit span detection, type assignment, and token-level tagging, using annotated examples for demonstration. Zero-shot version omits the examples.

Self-Refine Prompt (Few-Shot Excerpt)

```
<task>
<test_sentence>
{NEPALI SENTENCE}
</test_sentence>
Inside <reasoning>, follow a three-phase Self-Refine
      protocol:
  1. <phase>Draft</phase> - Create an initial BIO tag for
       each token.
  2. <phase>Critique</phase> - Examine the draft; point out
       BIO-format errors
     (e.g., an I-tag without a preceding B-tag) or wrong
          entity types.
  3. <phase>Repair</phase> - Produce a corrected tag list.

After finishing all three phases in <reasoning>, output
      only the repaired
BIO labels inside <tagged_output>.
</task>
<output_formatting>
The model must respond with exactly two root-level XML
      sections in order:
1. <reasoning> - all intermediate thinking (any structure
      you like).
2. <tagged_output> - one <pair> per word:
   <pair><word>WORD</word><pred_tag>TAG</pred_tag></pair>
Nothing is allowed outside these two tags.
</output_formatting>
```

Figure 5: Prompt template for **Few-Shot + Self-Refine** showing only parts different from Baseline. The model executes draft, critique, and repair steps, then outputs the final structured prediction. The zero-shot version omits annotated examples.

templates for all techniques mirror their few-shot versions, differing only in the absence of annotated training examples.

For the zero-shot configuration, prompts across all strategies follow a unified framework: defining the model's role, describing the BIO tagging scheme, and specifying strict output formatting. The key differences lie in how each strategy structures its reasoning—direct prediction (Baseline), stepwise justification (Chain-of-Thought), iterative self-correction (Self-Refine), or hierarchical decomposition (Least-to-Most).

In the few-shot setting, annotated examples are provided before the test sentence, but the overall output format and reasoning protocols of each strategy remain consistent with their zero-shot counterparts. Thus, while the Baseline approach tags directly from examples, the other methods expand upon this with increasingly explicit reasoning or multi-phase processes, leveraging observed patterns for improved prediction.

4 Experimental Setup

4.1 Dataset Preparation

For the purposes of this study, the Nepali Named Entity Recognition (NER) dataset introduced by Singh et al. (2019) was employed. This corpus consists of annotated Nepali text, with named entities categorized into three types: Person (PER), Location (LOC), and Organization (ORG), following the CoNLL-2003 tagging convention. The original dataset comprises a total of 3,289 sentences, of which 2,796 sentences were kept as examples for use in the few-shot configuration, while 493 sentences constituted the test set.

In this study, we selected up to 75 sentences as few-shot examples to guide model predictions. The distribution of entity tags within these examples, as the number of selected sentences increases, is summarized in Table 1. To further assess the impact of example selection, we performed a side-by-side comparison of two strategies: Random Sentence Selection, where example sentences are drawn at random, and our proposed Sentence Selection strategy, applied to a representative subset of 25 test sentences. The resulting tag distributions for each approach are reported in Table 2.

Sentences	B-ORG	I-ORG	B-PER	I-PER	B-LOC	I-LOC
1	1	2	8	7	7	1
5	9	20	17	16	15	6
10	20	32	31	35	24	10
15	37	59	55	59	40	12
20	54	85	71	75	50	12
25	67	102	82	88	63	12
50	106	167	131	137	105	16
75	136	208	167	174	140	19

Table 1: Tag counts across few-shot example sizes.

(a) Random Sentence Selection

# Sent.	B-ORG	I-ORG	B-PER	I-PER	B-LOC	I-LOC	O
5	4	1	3	2	1	0	79
10	9	14	6	3	4	0	267
25	17	14	26	16	14	0	477
50	55	56	43	27	30	2	1193
75	43	22	72	51	70	6	1959

(b) Sentence Selection Strategy

# Sent.	B-ORG	I-ORG	B-PER	I-PER	B-LOC	I-LOC	O
5	9	20	17	16	15	6	250
10	20	32	31	35	24	10	391
25	67	102	82	88	63	12	864
50	106	167	131	137	105	16	1665
75	136	208	167	174	140	19	2367

Table 2: NER tag counts across different dataset sizes using (a) random sentence selection and (b) sentence selection strategy.

4.2 Large Language Model

All experiments were conducted using Meta's quantized multilingual LLaMA 3.3:70B model (Grattafiori et al., 2024), a state-of-the-art open-source large language model accessible through the Ollama platform[1]. The model supports This transformer-based architecture has been pre-trained on more than 15 trillion high-quality tokens and supports a vocabulary of 128,000 tokens. The model leverages Grouped Query Attention (GQA) to improve inference efficiency. To address memory and computational limitations, the quantized Q4_K_M variant was utilized in all experiments. Model inference and evaluation were performed on a computing environment equipped with NVIDIA A100-SXM4-80GB GPUs (NVIDIA Corporation, 2025). The specific parameter settings employed throughout the experiments are detailed in Table 3.

Parameter	Model	Temp.	Top-p	Top-k	Seed
Value	LLaMA3.3:70B	0.7	0.9	50	23

Table 3: Generation parameters used for prompting Meta's quantized LLaMA 3.3:70B (Q4_K_M) model.

4.3 Prompting configurations

Setting	**Example Count**	**Strategies**
Zero-shot	0	All
Few-shot	1, 5, 10, 15, 20, 25, 50, 75	All

Table 4: Prompting configurations evaluated: all four strategies (*Baseline*, *Chain-of-Thought*, *Self-Refine*, *Least-to-Most*) in zero-shot and few-shot settings with varying in-context examples.

The prompting configurations explored in this study are summarized in Table 4. All four prompt engineering strategies—Baseline, Chain-of-Thought, Self-Refine, and Least-to-Most—were systematically assessed under both zero-shot and few-shot conditions. In the zero-shot configuration, the model received only the prompt template without any annotated examples (*Example Count: 0*). For the few-shot setting, the number of in-context annotated examples was progressively varied, with experiments conducted using 1, 5, 10, 15, 20, 25, 50, and 75 example sentences within each prompt. This experimental design enables a comprehensive investigation into the effects of prompt structure and the degree of in-context supervision on NER performance in a low-resource language setting.

4.4 Evaluation Metrics

Model performance was evaluated using standard metrics: *precision*, defined as the ratio of true positives to the sum of true positives and false positives; *recall*, the ratio of true positives to the sum of true positives and false negatives; and *F1-score*, computed as the harmonic mean of precision and recall (Hastie et al., 2016).

5 Results and Discussion

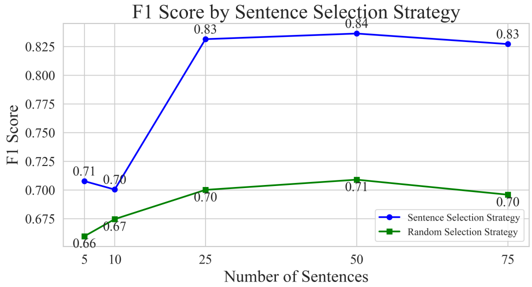


Figure 6: F1 scores on 25 test sentences, comparing the sentence selection strategy and random selection for few-shot prompting across 5, 10, 25, 50, and 75 examples.

Figure 6 highlights the comparative effectiveness of the sentence selection strategy, which consistently surpasses random selection at every evaluated example count. With only 5 annotated examples, sentence selection yields an F1 score of 0.71 compared to 0.66 for random selection. This performance gap widens with additional examples, reaching 0.83 versus 0.70 at 25 examples and peaking at 0.84 compared to 0.71 at 50 examples. These findings reinforce the critical importance of curating diverse, entity-rich examples for maximizing few-shot performance in Nepali NER.

Table 5 shows that retries were frequently necessary in zero-shot and one-shot scenarios, with many test sentences requiring multiple attempts and those reaching a third try not yielding valid outputs. In contrast, few-shot configurations with five or more annotated examples almost always produced valid responses on the first attempt, underscoring the stabilizing effect of minimal in-context supervision. As a result, zero-shot and one-shot settings were excluded from quantitative analysis, as they often led to hallucinated or unstructured outputs (see Figure 7), illustrating the limitations of cross-lingual transfer without explicit Nepali examples.

Of the prompting strategies evaluated, the Least-to-Most approach achieves the highest performance, attaining a peak F1 score of 0.8403 with 10 annotated examples and maintaining values above 0.79 as more examples are included (Figure 8). Both Self-Refine and Baseline strategies offer com-

[1] https://ollama.com/library/llama3.3:70b

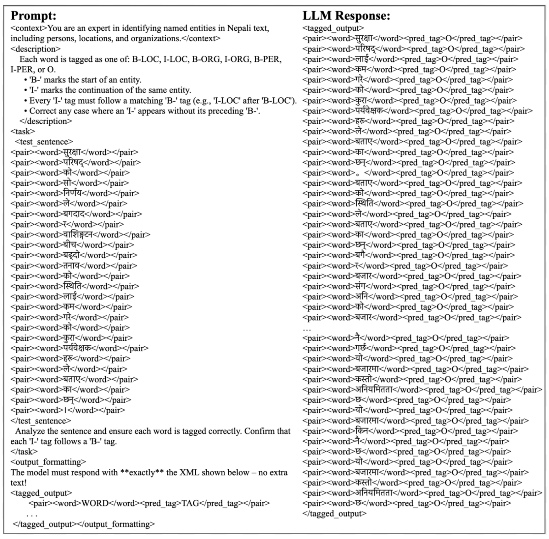
Technique	Num of Examples	Attempt	Test Sentences
Self-Refine	Zero-shot	1	96
		2	36
		3	**361**
	1	1	429
		2	26
		3	**38**
Baseline	Zero-shot	1	134
		2	42
		3	**317**
	1	1	482
		2	0
		3	**11**
Least-to-Most	Zero-shot	1	179
		2	43
		3	**271**
	1	1	484
		2	4
		3	**5**
Chain-of-Thought	Zero-shot	1	134
		2	42
		3	**317**
	1	1	346
		2	41
		3	**106**

Table 5: Retry attempt statistics for zero-shot and one-shot settings across all prompting strategies. The table reports the number of test sentences requiring each attempt to obtain an expected response. Notably, for all sentences that reached the third attempt, none yielded valid response, indicating that retries beyond the second attempt were not helpful.

petitive results, each exceeding an F1 score of 0.82 with moderate supervision. In contrast, the Chain-of-Thought approach lags behind, peaking at an F1 of 0.7998 and declining with additional examples. These trends are mirrored in the precision and recall plots (Figures 9 and 10) too.

Hence, both the zero-shot and one-shot configurations were excluded from the quantitative analysis, as they frequently resulted in hallucinated predictions (see Figure 7) or failed to produce outputs in the required format. In these settings, the Hindi-capable model, without explicit Nepali supervision, failed to generalize and routinely generated incomplete or unstructured predictions, underscoring the limitations of cross-lingual transfer without in-context examples.

A closer analysis of prediction errors reveals that most inaccuracies are concentrated in the I-LOC and I-ORG entity types, which persistently exhibit the lowest precision and recall. Such errors are especially pronounced in sentences with multi-token entities or morphologically complex phrases, where continuation tags are frequently misassigned, resulting in fragmented entity spans or BIO violations. By contrast, person entities (B-PER and I-PER) are recognized with consistently high precision and recall, underscoring their robustness across all prompting methods. Maintaining entity span continuity and disambiguating among similar entity types in complex, entity-dense contexts remain the principal challenges for prompt-based Nepali NER.



Prompt:

```
<context>You are an expert in identifying named entities in Nepali text,
including persons, locations, and organizations.</context>
<description>
  Each word is tagged as one of: B-LOC, I-LOC, B-ORG, I-ORG, B-PER,
I-PER, or O.
     • 'B-' marks the start of an entity.
     • 'I-' marks the continuation of the same entity.
     • Every 'I-' tag must follow a matching 'B-' tag (e.g., 'I-LOC' after 'B-LOC').
     • Correct any case where an 'I-' appears without its preceding 'B-'.
  </description>
<task>
 <test_sentence>
<pair><word>सुरक्षा</word></pair>
<pair><word>परिषद्</word></pair>
<pair><word>को</word></pair>
<pair><word>सो</word></pair>
<pair><word>निर्णय</word></pair>
<pair><word>ले</word></pair>
<pair><word>बगदाद</word></pair>
<pair><word>र</word></pair>
<pair><word>वाशिङ्गटन</word></pair>
<pair><word>बीच</word></pair>
<pair><word>बढ्दो</word></pair>
<pair><word>तनाव</word></pair>
<pair><word>को</word></pair>
<pair><word>स्थिति</word></pair>
<pair><word>लाई</word></pair>
<pair><word>कम</word></pair>
<pair><word>गरे</word></pair>
<pair><word>को</word></pair>
<pair><word>कुरा</word></pair>
<pair><word>पर्यवेक्षक</word></pair>
<pair><word>हरु</word></pair>
<pair><word>ले</word></pair>
<pair><word>बताए</word></pair>
<pair><word>का</word></pair>
<pair><word>छन्</word></pair>
<pair><word>।</word></pair>
</test_sentence>
 Analyze the sentence and ensure each word is tagged correctly. Confirm that
each 'I-' tag follows a 'B-' tag.
</task>
<output_formatting>
The model must respond with **exactly** the XML shown below – no extra
text!
<tagged_output>
     <pair><word>WORD</word><pred_tag>TAG</pred_tag></pair>
     . . .
 </tagged_output></output_formatting>
```

LLM Response:

```
<tagged_output>
<pair><word>सुरक्षा</word><pred_tag>O</pred_tag></pair>
<pair><word>परिषद्</word><pred_tag>O</pred_tag></pair>
<pair><word>लाई</word><pred_tag>O</pred_tag></pair>
<pair><word>कम</word><pred_tag>O</pred_tag></pair>
<pair><word>गरे</word><pred_tag>O</pred_tag></pair>
<pair><word>को</word><pred_tag>O</pred_tag></pair>
<pair><word>कुरा</word><pred_tag>O</pred_tag></pair>
<pair><word>पर्यवेक्षक</word><pred_tag>O</pred_tag></pair>
<pair><word>हरु</word><pred_tag>O</pred_tag></pair>
<pair><word>ले</word><pred_tag>O</pred_tag></pair>
<pair><word>बताए</word><pred_tag>O</pred_tag></pair>
<pair><word>का</word><pred_tag>O</pred_tag></pair>
<pair><word>छन्</word><pred_tag>O</pred_tag></pair>
<pair><word>。</word><pred_tag>O</pred_tag></pair>
<pair><word>बताए</word><pred_tag>O</pred_tag></pair>
<pair><word>को</word><pred_tag>O</pred_tag></pair>
<pair><word>स्थिति</word><pred_tag>O</pred_tag></pair>
<pair><word>ले</word><pred_tag>O</pred_tag></pair>
<pair><word>बताए</word><pred_tag>O</pred_tag></pair>
<pair><word>का</word><pred_tag>O</pred_tag></pair>
<pair><word>छन्</word><pred_tag>O</pred_tag></pair>
<pair><word>बगै</word><pred_tag>O</pred_tag></pair>
<pair><word>र</word><pred_tag>O</pred_tag></pair>
<pair><word>बजार</word><pred_tag>O</pred_tag></pair>
<pair><word>संग</word><pred_tag>O</pred_tag></pair>
<pair><word>अनि</word><pred_tag>O</pred_tag></pair>
<pair><word>को</word><pred_tag>O</pred_tag></pair>
<pair><word>बजार</word><pred_tag>O</pred_tag></pair>
…
<pair><word>नै</word><pred_tag>O</pred_tag></pair>
<pair><word>गर्छ</word><pred_tag>O</pred_tag></pair>
<pair><word>यो</word><pred_tag>O</pred_tag></pair>
<pair><word>बजारमा</word><pred_tag>O</pred_tag></pair>
<pair><word>कस्तो</word><pred_tag>O</pred_tag></pair>
<pair><word>अनियमितता</word><pred_tag>O</pred_tag></pair>
<pair><word>छ</word><pred_tag>O</pred_tag></pair>
<pair><word>यो</word><pred_tag>O</pred_tag></pair>
<pair><word>बजारमा</word><pred_tag>O</pred_tag></pair>
<pair><word>किन</word><pred_tag>O</pred_tag></pair>
<pair><word>नै</word><pred_tag>O</pred_tag></pair>
<pair><word>छ</word><pred_tag>O</pred_tag></pair>
<pair><word>यो</word><pred_tag>O</pred_tag></pair>
<pair><word>बजारमा</word><pred_tag>O</pred_tag></pair>
<pair><word>कस्तो</word><pred_tag>O</pred_tag></pair>
<pair><word>अनियमितता</word><pred_tag>O</pred_tag></pair>
<pair><word>छ</word><pred_tag>O</pred_tag></pair>
</tagged_output>
```

Figure 7: Example of zero-shot prompt and corresponding LLM response. The model hallucinated several tokens not present in the input prompt. For brevity, only part of the output is shown; ellipses (. . .) indicate omitted lines.

While our best prompt-based performance (F1 = 0.84, Least-to-Most) demonstrates the promise of prompt-only adaptation for low-resource NER, it is important to note that these results fall substantially short of the F1-scores routinely reported for fully supervised neural and transformer-based models on the same dataset, which regularly exceed 0.97 with fine-tuning (Table 6). This gap underscores both the progress and the inherent limitations of prompt-only methods in cross-lingual, low-resource settings.

Model	F1-Score
Stanford CRF	0.752
BiLSTM	0.847
BiLSTM + POS	0.836
BiLSTM + CNN (C)	0.864
BiLSTM + CNN (G)	0.867
BiLSTM + CNN (C) + POS	0.854
BiLSTM + CNN (G) + POS	0.855
MuRIL	0.979
BERT Multilingual	0.974
DistilBERT Multilingual	0.972
RemBERT	0.973

Table 6: F1-score of various traditional and BERT-based models on the dataset (Singh et al., 2019; Yadav et al., 2024).

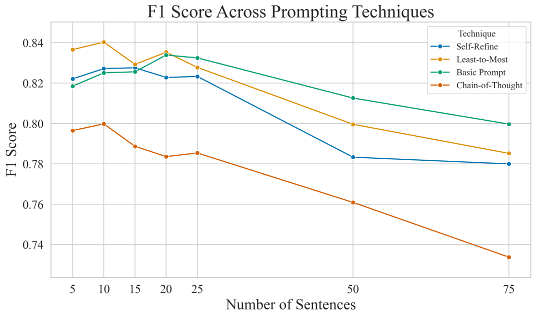


Figure 8: F1 score plot for all 493 test sentences using 5, 10, 25, 50, and 75 examples selected by the sentence selection strategy.

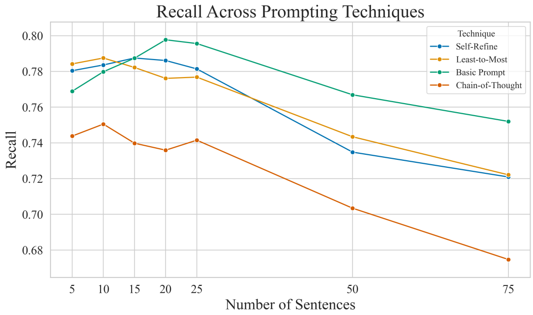


Figure 10: Recall score plot for all 493 test sentences using 5, 10, 25, 50, and 75 examples selected by the sentence selection strategy.

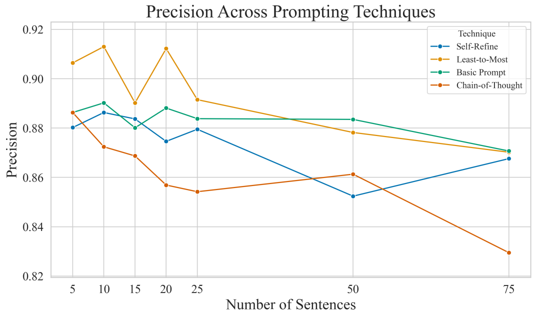


Figure 9: Precision score plot for all 493 test sentences using 5, 10, 25, 50, and 75 examples selected by the sentence selection strategy.

6 Conclusion

This work provides a comprehensive analysis of prompt engineering strategies for Nepali Named Entity Recognition (NER) using a Hindi-capable large language model, focusing on both zero-shot and few-shot scenarios. Through a systematic comparison of Baseline, Chain-of-Thought, Self-Refine, and Least-to-Most prompting techniques, we demonstrate that prompt-based adaptation—when supplied with as few as five well-chosen annotated examples—can achieve competitive F1 scores, peaking at 0.84 with the Least-to-Most approach. Our results reveal three central insights.

First, prompt-only cross-lingual transfer, in the complete absence of Nepali supervision (i.e., zero-shot and one-shot), fails to produce reliable or conforming outputs, with the model frequently hallucinating entities. This finding underscores the limitations of relying solely on linguistic similarity for zero-shot learning in truly low-resource settings.

Second, the introduction of minimal in-context supervision produces substantial gains in both output validity and overall NER performance. Among the prompting techniques evaluated, the Least-to-Most approach consistently yielded the highest scores, benefiting from its explicit, stepwise decomposition of the NER task. Both Self-Refine and Baseline strategies also proved effective, while Chain-of-Thought prompting lagged behind—suggesting that, for morphologically rich and structurally complex languages like Nepali, decomposition and error correction may be more beneficial than pure reasoning.

Third, the design and selection of prompt examples play a decisive role in few-shot settings. Our sentence selection strategy, prioritizing entity diversity and tag coverage, reliably outperformed random sampling, highlighting the importance of qualitative curation in low-resource prompt design.

While prompt-based methods offer a promising alternative to full model fine-tuning—enabling rapid adaptation with minimal supervision—there remains a pronounced performance gap relative to state-of-the-art transformer models trained with extensive Nepali data. Bridging this gap will require further innovations in prompt engineering, possibly in combination with parameter-efficient tuning or retrieval-augmented methods.

Future research should extend these insights to a broader range of low-resource languages and entity categories, investigate the integration of external linguistic resources, and explore adaptive prompting schemes that further minimize supervision while maximizing generalization. Our findings underscore both the potential and the current boundaries of prompt-based NER in cross-lingual, low-resource scenarios, paving the way for more robust, accessible solutions.

References

Anthropic. 2025. Prompt engineering overview. Accessed: 2025-05-15.

S. Bam and T. Shahi. 2014. Named entity recognition for nepali text using support vector machines. *Intelligent Information Management*, 6(2):21–29.

Vincent Beaufils. 2015–2025. elinguistics.net: Quantifying the genetic proximity between languages.

Maddineni Bhargava, Karthika Vijayan, Oshin Anand, and Gaurav Raina. 2023. Exploration of transfer learning capability of multilingual models for text classification. In *Proceedings of the 2023 5th International Conference on Pattern Recognition and Intelligent Systems*, PRIS '23, page 45–50, New York, NY, USA. Association for Computing Machinery.

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Qi Cheng, Liqiong Chen, Zhixing Hu, Juan Tang, Qiang Xu, and Binbin Ning. 2024. A novel prompting method for few-shot ner via llms. *Natural Language Processing Journal*, 8:100099.

Abolfazl Farahani, Behrouz Pourshojae, Khaled Rasheed, and Hamid R. Arabnia. 2021. A concise review of transfer learning. *CoRR*, abs/2104.02144.

Xiaocheng Feng, Xiachong Feng, Bing Qin, Zhangyin Feng, and Ting Liu. 2018. Improving low resource named entity recognition using cross-lingual knowledge transfer. In *IJCAI*, volume 1, pages 4071–4077.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2016. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edition. Springer, New York.

Gaurav Kamath and Sowmya Vajjala. 2025. Does synthetic data help named entity recognition for low-resource languages?

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE transactions on knowledge and data engineering*, 34(1):50–70.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*.

Pengfei Liu, Weizhe Yuan, Jinlan Fu, Zhengbao Jiang, Hiroaki Hayashi, and Graham Neubig. 2021. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2021. Template-free prompt tuning for few-shot ner. *arXiv preprint arXiv:2109.13532*.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback.

Gopal Maharjan, Bal Krishna Bal, and Santosh Regmi. 2019. Named entity recognition (ner) for nepali. In *Creativity in Intelligent Technologies and Data Science: Third Conference, CIT&DS 2019, Volgograd, Russia, September 16–19, 2019, Proceedings, Part II 3*, pages 71–80. Springer.

Rajesh Kumar Mundotiya, Shantanu Kumar, Ajeet kumar, Umesh Chandra Chaudhary, Supriya Chauhan, Swasti Mishra, Praveen Gatla, and Anil Kumar Singh. 2020. Development of a dataset and a deep learning baseline named entity recognizer for three low resource languages: Bhojpuri, maithili and magahi.

Rudra Murthy, Mitesh M Khapra, and Pushpak Bhattacharyya. 2018. Improving ner tagging performance in low-resource languages via multilingual learning. *ACM Transactions on Asian and Low-Resource Language Information Processing (TALLIP)*, 18(2):1–20.

Marco Naguib, Xavier Tannier, and Aurélie Névéol. 2024. Few shot clinical entity recognition in three languages: Masked language models outperform llm prompting. *arXiv preprint arXiv:2402.12801*.

Nobal Niraula and Jeevan Chapagain. 2023. Danfener - named entity recognition in nepali tweets. *The International FLAIRS Conference Proceedings*, 36(1).

NVIDIA Corporation. 2025. Nvidia a100. Whitepaper, NVIDIA Corporation.

Libo Qin, Qiguang Chen, Yuhang Zhou, Zhi Chen, Yinghui Li, Lizi Liao, Min Li, Wanxiang Che, and Philip S. Yu. 2024. Multilingual large language model: A survey of resources, taxonomy and frontiers.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. 2019. A survey of cross-lingual word embedding models. *Journal of Artificial Intelligence Research*, 65:569–631.

Timo Schick and Hinrich Schütze. 2020. It's not just size that matters: Small language models are also few-shot learners. *arXiv preprint arXiv:2009.07118*.

Oyesh Mann Singh, Ankur Padia, and Anupam Joshi. 2019. Named entity recognition for nepali language. In *2019 IEEE 5th international conference on collaboration and internet computing (cic)*, pages 184–190. IEEE.

S Sivarajkumar, M Kelley, A Samolyk-Mazzanti, S Visweswaran, and Y Wang. 2024. An empirical evaluation of prompting strategies for large language models in zero-shot clinical natural language processing: Algorithm development and validation study. *JMIR Medical Informatics*, 12:e55318.

Shaden Smith, Mostofa Patwary, Brandon Norick, Patrick LeGresley, Samyam Rajbhandari, Jared Casper, Zhun Liu, Shrimai Prabhumoye, George Zerveas, Vijay Korthikanti, et al. 2022. Using deepspeed and megatron to train megatron-turing nlg 530b, a large-scale generative language model. *arXiv preprint arXiv:2201.11990*.

Bipesh Subedi, Sunil Regmi, Bal Krishna Bal, and Praveen Acharya. 2024. Exploring the potential of large language models (LLMs) for low-resource languages: A study on named-entity recognition (NER) and part-of-speech (POS) tagging for Nepali language. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 6974–6979, Torino, Italia. ELRA and ICCL.

Prajwal Thapa, Jinu Nyachhyon, Mridul Sharma, and Bal Krishna Bal. 2025. Development of pre-trained transformer-based models for the Nepali language. In *Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL 2025)*, pages 9–16, Abu Dhabi, UAE. International Committee on Computational Linguistics.

David Vilar, Markus Freitag, Colin Cherry, Jiaming Luo, Viresh Ratnakar, and George Foster. 2022. Prompting palm for translation: Assessing strategies and performance. *arXiv preprint arXiv:2211.09102*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903.

Dipendra Yadav, Sumaiya Suravee, Tobias Strauß, and Kristina Yordanova. 2024. Cross-lingual named entity recognition for low-resource languages: A Hindi-Nepali case study using multilingual BERT models. In *Proceedings of the Fourth Workshop on Multilingual Representation Learning (MRL 2024)*, pages 167–174, Miami, Florida, USA. Association for Computational Linguistics.

Jin Zhang, Fan Gao, Linyu Li, Yongbin Yu, Xiangxiang Wang, Nyima Tashi, and Gadeng Luosang. 2025. Retrieveall: A multilingual named entity recognition framework with large language models.

Denny Zhou, Nathanael Schärli, Le Hou, Jason Wei, Nathan Scales, Xuezhi Wang, Dale Schuurmans, Claire Cui, Olivier Bousquet, Quoc Le, and Ed Chi. 2023. Least-to-most prompting enables complex reasoning in large language models.