COMI-LINGUA: Expert Annotated Large-Scale Dataset for Multitask NLP in Hindi-English Code-Mixing

Rajvee Sheth, Himanshu Beniwal, Mayank Singh

LINGO Research Group, Indian Institute of Technology Gandhinagar, India

Correspondence: lingo@iitgn.ac.in

Abstract

We introduce COMI-LINGUA, the largest manually annotated Hindi-English code-mixed dataset, comprising 125K+ high-quality instances across five core NLP tasks: Token-level Language Identification, Matrix Language Identification, Named Entity Recognition, Part-Of-Speech Tagging and Machine Translation. Each instance is annotated by three bilingual annotators, yielding over 376K expert annotations with strong inter-annotator agreement (Fleiss' Kappa $\geq$ 0.81). The rigorously preprocessed and filtered dataset covers both Devanagari and Roman scripts and spans diverse domains, ensuring real-world linguistic coverage. Evaluation reveals that closed-weight LLMs significantly outperform traditional tools and open-weight models in zero-shot settings. Notably, one-shot prompting consistently boosts performance across tasks, especially in structure-sensitive predictions like POS and NER. Fine-tuning open-weight LLMs on COMI-LINGUA demonstrates substantial improvements, achieving up to 95.25 F1 in NER, 98.77 F1 in MLI, and competitive MT performance, setting new benchmarks for Hinglish code-mixed text. COMI-LINGUA is publicly available at this URL[1].

1 Introduction

Code-mixing is the blending of multiple languages within a single utterance—a pervasive phenomenon in multilingual societies, especially on social media platforms (Jamatia et al., 2020; Srivastava and Singh, 2020). Over half of the world's population is bilingual or multilingual and frequently uses mixed-language expressions in digital communication (Grosjean, 2021). In the Indian context, Hindi-English (Hinglish) code-mixed text is particularly widespread and presents significant

computational challenges due to orthographic complexity, frequent language switches, and script variation between Devanagari and Roman forms (Bali et al., 2014; Takawane et al., 2023; Thara and Poornachandran, 2018). A characteristic example is: Kal mujhe ऑफिस जाना hai, but ट्राफिक will be an issue, where Hindi and English tokens co-occur and certain English words like "office" and "traffic" may appear in Devanagari script. (English Translation: "Tomorrow I have to go to the office, but traffic will be an issue.").

Despite growing interest, current Hinglish datasets have critical limitations: *(1)* a predominant focus on Roman script, ignoring natural script variation (Begum et al., 2016; Bali et al., 2014; Srivastava et al., 2020), *(2)* limited scale and coverage (Srivastava and Singh, 2021a; Kumar et al., 2018; Tiwari et al., 2024; Kartik et al., 2024), *(3)* insufficient task diversity (Aguilar et al., 2020; Khanujaa et al., 2020; Bohra et al., 2018; Khanuja et al., 2020), and *(4)* reliance on synthetic data generation and labeling rather than human annotation (Chatterjee et al., 2022; Srivastava and Singh, 2021c; Kartik et al., 2024; Sravani and Mamidi, 2023).

To address these limitations, we present a novel comprehensive dataset **COMI-LINGUA** (**CO**de-**MI**xing and **LING**uistic Insights on Natural Hinglish **U**sage and **A**nnotation) that advances Hinglish code-mixing research. The key contributions include:

- Curation of the largest publicly available Hinglish dataset (376K manually annotated instances), released under a CC-BY-4.0 license, capturing real-world code-mixing behavior across both Roman and Devanagari scripts. Each instance is annotated by one annotator across one of the key NLP tasks: token-level language identification, matrix language identification, named entity recognition, part-of-speech tagging and machine

---

[1] https://huggingface.co/datasets/LingoIITGN/ COMI-LINGUA Version 1.0, Updated till 15nd September 2025.

Annotation Tasks

Token-level Language Identification
Input sentence: Made in इंडिया ki ना केवल ग्लोबल डिमांड ho , बल्कि ग्लोबल acceptance ho हमे ये सुनिश्चित karna hai .
Language Tags: en en en hi hi hi en en hi ot hi en   en   hi hi hi hi   hi hi ot
English Translation: Made in India should not only have global demand but also global acceptance; we must ensure this.

Matrix Language Identification
Input sentence: SBI ने चालू कारोबारी साल में UP में 550 crore में रुपए का गोल्ड लोन देने का लक्ष्य रखा है।
Matrix Language: hi
English Translation: SBI has set a target of disbursing gold loans worth 550 crore rupees in UP in the current financial year.

Named Entity Recognition
Input sentence: Kendra Spade एक American Film Actress हैं और इनका जन्म 11 May 1998 को New Orleans, United States में हुआ था।
Named Entities: 'Kendra Spade: Person, '11 May 1998': DATE, 'New Orleans': Location, 'United States': GPE.
English Translation: Kendra Spade is an American film actress, and she was born on May 11, 1998, in New Orleans, United States.

Part-of-Speech Tagging
Input sentence: आज भारत FDI Confidence Index में top two emerging market performers में से एक है ।
POS Tags: ADV PROPN PROPN NOUN NOUN ADP ADJ NUM ADJ ADJ NOUN ADP DET NUM VERB X
English Translation: Mirabai Chanu won the first gold medal for India in the 21st Commonwealth Games.

Machine Translation
Input sentence: Indian Air Force ने मौसम वैज्ञानिक भर्ती करने के लिए result जारी कर दिया है।
Standard English Translation: The Indian Air Force has released the results for the recruitment of Meteorological Scientists.
Romanized Hindi Translation: Bhartiya Vayu Sena ne mausam vigyaani bharti ke result jaari kar diya hain.
Devanagari Hindi Translation: भारतीय वायु सेना ने मौसम विज्ञानी भर्ती के करने के लिए परिणाम जारी कर दिया है।
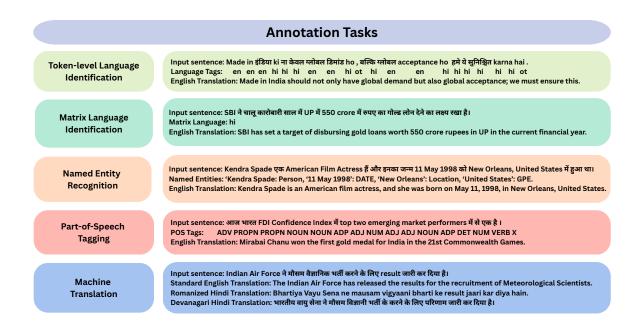
Figure 1: Sample Annotations Across COMI-LINGUA Tasks: Shown here are annotated instances for each of the five tasks defined in the COMI-LINGUA task set, emphasizing the annotation strategy and linguistic diversity.

translation.

- Robust benchmarking of state-of-the-art multilingual LLMs (mLLMs), including both open-weight and closed-weight models—alongside traditional NLP tools, under two inference paradigms: zero-shot and one-shot in-context learning.

- In-depth error analysis of mLLMs on code-mixed tasks, uncovering critical limitations such as misclassification of English borrowings in Devanagari script, context truncation, overfitting in one-shot settings, prompt mimicry, repetitive or hallucinated outputs, and practical deployment barriers like API usage constraints—highlighting persistent challenges in script-aware and context-sensitive language modeling.

2 Related Work

Code-mixing—the blending of multiple languages in a single utterance—poses major challenges for NLP due to its structural variability (Srivastava and Singh, 2021a). This is especially true for Hinglish, given their distinct scripts and syntax (Bali et al., 2014). Progress is hindered by the lack of large, annotated datasets, as collecting and labeling such data remains costly and labor-intensive (Srivastava and Singh, 2021a).

Language Identification is a foundational task in code-mixed NLP. Multiple approaches have been developed to detect language boundaries within mixed-language sequences, including statistical models, CRFs, and deep learning-based techniques (Shekhar et al., 2020; Singh et al., 2018a; Gundapu and Mamidi, 2018; Molina et al., 2016). These efforts have paved the way for improved preprocessing and downstream modeling of code-mixed data.

Named Entity Recognition in code-mixed text has seen significant progress through both resource development and model improvements. Dowlagar and Mamidi (2022) showed that leveraging multilingual data enhances NER accuracy, while Ansari et al. (2019) created cross-script datasets using Wikipedia. Transformer-based approaches and meta-embeddings have also been effective in improving NER for Indian code-mixed data (Priyadharshini et al., 2020).

Part-of-Speech Tagging A variety of annotated datasets have been introduced for POS tagging in code-mixed contexts. Singh et al. (2018b) and Vyas et al. (2014) developed corpora from Twitter and Facebook, respectively, while Pratapa et al. (2018) generated synthetic datasets for evaluating bilingual word embeddings. Sequiera et al. (2015) experimented with various machine learning algorithms, and Chatterjee et al. (2022) introduced PACMAN, a large-scale synthetic POS-tagged dataset that achieved state-of-the-art performance in code-mixed POS tagging tasks.

Machine Translation for code-mixed content re-

mains a growing research area. Dhar et al. (2018) and Srivastava and Singh (2020) developed parallel corpora for Hinglish code-mixed sentences, while Hegde and Lakshmaiah (2022) proposed translation models using transliteration and pseudo-translation, achieving competitive results in the MixMT shared task at WMT 2022.

Benchmarking and Evaluation Frameworks Several benchmark datasets have been introduced to evaluate NLP systems on code-mixed tasks. LinCE (Aguilar et al., 2020) provides a comprehensive benchmark covering 11 corpora and 4 language pairs. GLUECoS (Khanuja et al., 2020) demonstrated the benefits of fine-tuning multilingual models on code-switched datasets across multiple tasks. Emotion and sentiment annotation efforts, such as the Hinglish Twitter corpus by Vijay et al. (2018), the L3Cube-HingCorpus (Nayak and Joshi, 2022), and the emotion-annotated SentiMix dataset by Ghosh et al. (2023), further support affective computing in code-mixed settings.

Despite ongoing efforts, standardized benchmarks for evaluating LLMs on diverse Hinglish code-mixed tasks—such as acceptability judgments, syntactic fluency, and translation fidelity—remain limited. Existing benchmarks are often narrow in scope and rely on synthetic or small-scale data. To address this, we curate the largest high-quality, human-annotated dataset for training and evaluating LLMs on a broad range of Hinglish code-mixed phenomena. It serves as both an evaluation suite and a diagnostic tool to advance multilingual and code-mixed language understanding research.

3 The COMI-LINGUA dataset

3.1 Raw Dataset Curation

We curated raw data from publicly accessible and licensed platforms spanning diverse categories such as news, politics, entertainment, social events, sports, and informational content, with a focus on the Indian subcontinent. Sources included prominent news portals and official digital archives, detailed in Appendix §A.1. The collected content was cleaned using regex-based preprocessing to remove noise such as advertisements, HTML tags, and footers, and then segmented into individual sentences. A Code-Mixing Index (CMI, Das and Gambäck (2014)) was computed for each sentence, and only those sentences with a CMI score ≥ 9 were retained to ensure a substantial degree of code-mixing. Given the under-representation of

mixed Devanagari-Roman script samples in existing datasets, we also collected supplementary data to enhance coverage and linguistic diversity. This includes enriching the dataset by incorporating additional Hinglish code-mixed samples from prior works (Srivastava and Singh, 2020; Gupta et al., 2023; Singh et al., 2018c) and from Hugging-Face[2].

3.2 Dataset Processing

The preprocessing pipeline was designed to enhance the quality and neutrality of the corpus through rigorous noise reduction techniques. To ensure the dataset was both clean and relevant, we removed duplicate instances, hate speech, and abusive content. Sentences containing offensive or inappropriate language were identified and filtered out using established profanity and hate speech detection tools, including *thisandagain*[3] and *Hate-Speech-Detection-in-Hindi*[4].

At the token level, additional preprocessing steps were applied. Sentences with fewer than five tokens were discarded to eliminate non-informative content such as fragments, abbreviations, emojis, and filler phrases—commonly arising from typing errors or social media discourse. Examples of such removed content include: *'#GuessTheSong'*, *'during dinner'*, and *'@enlightenedme bas ek hi'*. Further data refinement was conducted during the manual annotation process (see Section 3.4 for more details).

3.3 Data Annotation

To annotate the Hinglish code-mixed corpus, we employed COMMENTATOR (Sheth et al., 2024), a robust annotation framework specifically designed for multilingual code-mixed text.

The annotation was carried out by a team of three graduate-level experts proficient in both Hindi and English. All annotators possess prior experience with social media content and demonstrate strong programming capabilities, along with familiarity with using version control systems. These competencies contributed to a systematic, efficient, and reproducible annotation process. The annotators were recruited specifically for this

---

[2]https://huggingface.co/datasets/pardeep/youtube-vidoes-transcripts-hindi-english/
[3]https://github.com/thisandagain/washyourmouthoutwithsoap/blob/develop/README.md
[4]https://github.com/victorknox/Hate-Speech-Detection-in-Hindi/blob/main/README.md

project and were compensated at a rate of approximately $1.64 per hour. The funding for the annotation work was provided through a government-sponsored initiative; the compensation adheres to standard remuneration practices considered appropriate for the annotators' qualifications and the economic context of their country of residence.

We selected five diverse annotation tasks with a partially overlapping annotation structure across task instances, balancing well-established tasks with high reliability and underexplored challenges. Annotators followed detailed guidelines with examples to ensure consistency and clarity across tasks (Appendix §A.3, Figure 1). The tasks are:

1. **Token-level Language Identification (LID):** In this task, each token in the dataset was assigned one of three possible language labels: English (en), Hindi (hi), or Other (ot). Initial language tags were generated using Microsoft's Language Identification Tool[5], which served as a baseline for further manual refinement. As shown in Figure 1, each token is assigned a language tag.

2. **Matrix Language Identification (MLI):** Each sentence is annotated with a Matrix Language, which identifies the dominant language governing the grammatical structure of the sentence. In code-mixed text, even when multiple languages are interspersed, one language typically dictates the syntactic and morphosyntactic framework of the utterance. Figure 1 showcases a sentence annotated with its matrix language.

3. **Named Entity Recognition (NER):** In the NER task, each token in a sentence is annotated with a label from a predefined set of entity types outlined in Table 1. These include conventional categories, such as Person, Location, Organization, Date/Time, and GPE (Geo-Political Entities), as well as social media-specific types like Hashtags, Mentions, and emojis. An instance of annotated entities across different types is shown in Figure 1. This allows the annotation schema to comprehensively capture the diversity and informality observed in code-mixed social media text.

Entity Type	Description
Person	Names of individuals
Location	Non-political physical locations
Organization	Institutions or companies
Date/Time	Temporal expressions (e.g., dates)
GPE	Geo-Political Entities
Hashtags	Words prefixed by '#'
Mentions	User mentions prefixed by '@'
Emoji	Emoticons conveying emotions

Table 1: Named entity types and their descriptions in our annotation schema.

POS Tag	Description
NOUN	Common nouns
PROPN	Proper nouns
VERB	Verbs in all tenses and moods
ADJ	Adjectives describing nouns
ADV	Adverbs modifying verbs
ADP	Adpositions
PRON	Pronouns
DET	Determiners
CONJ	Conjunctions
PART	Grammatical particles
PRON_WH	Wh-pronouns
PART_NEG	Negative particles
NUM	Numerals and cardinal numbers
X	Typos, abbreviations, punctuation

Table 2: Part-of-speech tags and their descriptions used in our annotation schema.

4. **Part-of-Speech (POS) Tagging:** Each token in the code-mixed dataset was annotated with a Part-of-Speech (POS) tag selected from the **Universal POS tagset** proposed by Singh et al. (2018b). The tagset, summarized in Table 2, was chosen for its language-agnostic design, enabling consistent annotation of Hindi and English words in a single sentence—an essential feature for handling code-mixed content effectively. A representative example is presented in Figure 1. Initial predictions for POS tags were generated using the *CodeSwitch NLP* library[6], which supports multilingual code-mixed data and provides pre-trained models suitable for tagging noisy, informal text commonly found on social media platforms.

5. **Machine Translation (MT):** This task involves constructing parallel translations for code-mixed sentences into three distinct formats: (i) Standard English, (ii) Romanized Hindi and (iii) Devanagari Hindi. The goal is to facilitate a multilingual Hinglish sentence to align with its respective translations

---

[5]https://github.com/microsoft/LID-tool

[6]https://github.com/sagorbrur/codeswitch

across scripts and languages. A representative translation instance across the three formats is shown in Figure 1. Initial translation predictions were generated using the LLaMA 3.3 language model[7].

For all tasks, we used state-of-the-art NLP tools or LLMs for automated pre-annotation, generating initial labels based on task-specific criteria. Expert annotators then refined these outputs through manual post-editing. This two-stage process ensured high-quality annotations while improving consistency and speeding up dataset creation.

3.4 Manual Data Refinement

During the annotation phase, the dataset underwent iterative refinement to ensure quality and consistency, guided by annotator feedback on instances to be excluded (see Table 8 in Appendix §A.2). We removed sentences if they (i) were monolingual English or Hindi, (ii) lacked relevant linguistic tags or named entities, contained no meaningful content, or merged multiple instances into one, or (iii) included languages other than Hindi and English, which were beyond the scope of this study. This refinement process was crucial for preserving corpus integrity and ensuring that the final dataset consisted solely of high-quality Hinglish code-mixed text. The Raw and Filtered columns in Table 3 represent the number of original instances provided for initial annotation and the final number of instances retained after annotation, respectively. The difference between these values corresponds to instances flagged by annotators as not satisfying the manual annotation criteria.

3.5 Annotation Efforts and Quality

The manual annotation process involved substantial human effort across all tasks, particularly in refining the outputs of automated tools. For example, for the LID task, each annotator reviewed 504,102 tokens and flipped an average of 95,670 tokens—approximately 19% of the original predictions. In the POS task, 63,002 of 427,941 tokens were corrected, indicating a 15% flip rate. Similarly, for the NER task, each annotator modified about 98,760 out of 538,160 tokens, translating to 18% manual corrections. For the MLI task, no initial predictions were provided, leading to 100%

Task	Raw	Filtered	IAA	CMI
LID	29,950	25,772	0.834	20.87
MLI	29,950	25,772	0.976	20.87
NER	26,929	24,913	0.852	14.38
POS	27,229	24,598	0.817	21.60
MT	26,727	24,558	-	17.07
Total / Avg.	140,785	125,615	0.863	18.96

Table 3: Corpus Statistics: The Raw and Filtered columns represent the number of original instances provided for initial annotation and the final instances retained after annotation, respectively. **Note:** IAA was not computed for the MT task as it is a generative task.

of the sentences being annotated. To assess annotation reliability, we computed inter-annotator agreement (IAA) using Fleiss' Kappa (Fleiss, 1971), a standard metric for evaluating consistency among multiple annotators on categorical labels (Hallgren, 2012). All classification tasks achieved Fleiss' Kappa scores above **0.817**, indicating substantial to near-perfect agreement (Table 3). As machine translation is a generative task, IAA was not calculated. While not a direct measure of quality, the final dataset retains a high level of code-mixing, with an average CMI exceeding **14** across tasks, ensuring strong code-mixing.

The **COMI-LINGUA** consists of 125,615 high-quality instances spanning five tasks, each independently annotated by three expert annotators, yielding a total of 376,845 annotations (see Table 9). To our knowledge, it is the largest manually annotated code-mixed dataset to date. For each task, We provide two random splits: a test set of 5,000 instances and a training set comprising the remainder (as detailed in Table 9, Appendix §B). Zero- and one-shot prompting was evaluated only on the fixed test set, whereas fine-tuning was carried out on the training split, with performance reported on the same 5,000-instance test set.

4 Experiments

4.1 Baseline Tools and LLMs

We conducted a comprehensive evaluation of existing tools and language models on the COMI-LINGUA Benchmark. Our experimental setup spans traditional NLP toolkits, state-of-the-art open-weight LLMs, and proprietary commercial models. These systems are evaluated on their performance across five diverse Hinglish code-mixed NLP tasks, detailed in Section 3.3.

The traditional tools evaluated in this study include the Microsoft LID[5] for token-level lan-

Model/Library	LID			MLI			NER			POS		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Zero-shot												
claude-3.5-sonnet	92.8	92.4	92.1	98.8	83.5	90.0	59.1	55.1	56.7	75.3	64.8	69.0
gpt-4o	**92.8**	**92.8**	**92.7**	98.4	97.9	98.1	60.5	60.1	60.1	**76.1**	**66.0**	**70.1**
gemini-1.5-Flash	82.9	40.4	47.9	98.8	21.4	33.7	44.2	44.2	43.8	73.4	62.4	66.5
LLaMA-3.3-instruct	73.4	73.7	73.3	98.8	59.0	73.1	67.5	67.3	**66.8**	74.3	65.5	68.9
mistral-instruct	54.5	39.0	42.4	98.1	58.7	72.3	65.1	41.5	50.2	10.2	6.72	7.78
command-a-03-2025	92.0	92.0	91.8	**98.5**	**98.0**	**98.3**	65.9	67.8	66.6	73.5	65.4	68.6
codeswitch	-	-	-	-	-	-	81.6	83.1	81.2	**89.1**	**87.8**	**88.2**
Microsoft LID	80.2	76.5	74.4	-	-	-	-	-	-	-	-	-
One-shot												
claude-3.5-sonnet	93.0	92.7	92.5	**98.8**	**98.9**	**98.8**	**85.9**	**85.2**	**85.0**	81.4	79.2	79.3
gpt-4o	**93.9**	**94.0**	**93.8**	98.7	97.7	98.1	77.4	75.8	76.0	81.6	78.0	78.9
gemini-1.5-Flash	80.2	76.5	74.4	98.4	40.4	56.4	66.5	67.5	66.0	72.9	64.6	68.0
LLaMA-3.3-instruct	90.3	89.6	89.3	98.8	97.8	98.2	79.0	79.1	78.4	**85.1**	**84.0**	**84.1**
mistral-instruct	72.1	70.0	70.1	98.3	88.1	92.7	65.5	44.4	52.6	77.3	66.9	69.8
command-a-03-2025	92.1	91.7	91.3	98.9	98.7	98.3	76.7	78.9	77.3	74.5	65.7	69.5

Table 4: Performance metrics on the COMI-LINGUA test sets for various models across different experimental settings (Zero-shot, One-shot) and tasks: LID, MLI, NER, and POS tagging. P, R, and F_1 denote Precision, Recall, and F1-score respectively. '-' indicates that the task is not supported by the respective tool.

Model	En		RH		DH	
	B_{en}	cF_{en}	B_{rh}	cF_{rh}	B_{dh}	cF_{dh}
Zero-shot						
claude-3.5-sonnet	48.1	63.5	48.6	64.4	56.0	65.7
gpt-4o	28.8	42.4	27.5	41.7	32.0	41.8
gemini-1.5-Flash	48.2	66.0	28.9	50.4	56.9	66.8
LLaMA-3.3-instruct	55.4	71.4	50.5	68.1	59.8	71.5
mistral-instruct	23.5	49.9	5.4	25.3	18.1	40.7
command-a-03-2025	38.6	58.5	35.2	57.0	48.8	61.8
One-shot						
claude-3.5-sonnet	50.9	68.5	52.2	69.4	63.5	73.4
gpt-4o	50.2	68.6	50.2	68.4	58.4	70.1
gemini-1.5-Flash	39.5	57.6	40.5	58.7	58.9	69.6
LLaMA-3.3-instruct	62.2	74.8	54.8	71.3	60.7	70.3
mistral-instruct	30.0	53.8	19.6	44.1	18.5	40.5
command-a-03-2025	52.9	68.6	42.0	60.4	56.1	66.6

Table 5: MT performance metrics on the COMI-LINGUA test sets for various models across Zero-shot and One-shot settings. B_{en}, B_{dh}, and B_{rh} represent BLEU scores, and cF_{en}, cF_{dh}, and cF_{rh} represent chrF++ scores for Standard English, Devanagari Hindi, and Romanized Hindi translation outputs, respectively.

guage identification and the codeswitch toolkit[6] for POS and NER tasks in multilingual text, which provides a rule-based pipeline for annotating syntactic and semantic information in code-switched corpora. The four commercial closed-weight systems considered in our evaluation include: claude-3.5-Sonnet (Anthropic, 2024), gpt-4o (Achiam et al., 2023), gemini-1.5-Flash (Anil et al., 2023) and command-a-03-2025 (111B) (Co-

here et al., 2025). In addition, we assess open-weight LLMs llama-3.3-instruct (70B) (Touvron et al., 2023) and mistral-instruct (7B) (Jiang et al., 2023).

We create specific prompt templates for each task to generate accurate, task-aligned responses from LLMs. The prompt template includes a high-level description of the task, specific annotation or tagging rules, and illustrative examples wherever applicable. For each of our five tasks, we developed two prompt variants: a zero-shot version providing only task instructions and a one-shot version that includes a single demonstrative example with instructions. The prompts are presented as a system-level instruction, followed by the user-supplied test input (i.e., a code-mixed sentence or token sequence). The complete prompt template used for each task under each prompt variant is detailed in Appendix §B.

4.2 Evaluation Metrics

We employ a suite of standard evaluation metrics, appropriately chosen for each task's nature. For token-level classification tasks—LID, POS, and NER—we report Precision (P), Recall (R), and the F_1-score, computed at the macro level. For the MLI task, which is a sentence-level classification problem, we adopt the same classification metrics—P, R, and F_1—computed on a per-sentence basis. For

Model	LID			MLI			NER			POS		
	P	R	F_1	P	R	F_1	P	R	F_1	P	R	F_1
Zero-shot												
`LLaMA-3.1-8B-In`	**62.5**	**70.0**	**62.3**	96.8	4.6	8.9	**63.4**	**66.8**	**64.8**	57.8	19.7	26.8
`aya-expanse-8b`	51.0	70.5	59.0	**98.7**	**59.5**	**74.2**	54.4	68.2	59.8	**76.9**	29.5	**40.5**
`Qwen2.5-7B-In`	56.4	69.6	59.1	98.6	21.2	34.8	61.0	65.3	57.3	68.4	8.70	9.06
One-shot												
`LLaMA-3.1-8B-In`	**83.0**	**80.3**	**81.4**	98.5	59.5	74.1	72.5	67.3	68.5	**72.7**	**63.2**	**64.7**
`aya-expanse-8b`	73.0	71.0	70.4	98.3	**81.3**	**89.0**	**79.7**	**81.4**	**79.1**	55.2	48.7	48.2
`Qwen2.5-7B-In`	57.1	74.8	64.8	98.4	61.9	75.7	74.1	76.3	74.0	70.4	59.8	63.0
Fine-tuned												
`LLaMA-3.1-8B-In`	**95.2**	**94.5**	**94.7**	98.0	88.7	93.0	**95.2**	**95.2**	**95.2**	86.6	86.1	86.1
`aya-expanse-8b`	87.4	86.9	87.1	**98.6**	**98.9**	**98.7**	94.9	94.9	94.9	**88.9**	**88.5**	**88.6**
`Qwen2.5-7B-In`	84.4	80.2	81.7	97.8	93.8	95.8	94.3	94.3	94.2	89.0	88.5	88.6

Table 6: Performance metrics on the COMI-LINGUA test sets for three LLMs across different experimental settings (Zero-shot, One-shot, Fine-tuned) on four tasks: LID, MLI, NER, and POS tagging. Metrics shown are Precision (P), Recall (R), and F1-score (F_1). Abbreviations `LLaMA-3.1-8B-In` and `Qwen2.5-7B-In` denote `LLaMA-3.1-8B-Instruct` and `Qwen2.5-7B-Instruct` respectively.

Model	En		RH		DH	
	B_{en}	cF_{en}	B_{rh}	cF_{rh}	B_{dh}	cF_{dh}
Zero-shot						
`LLaMA-3.1-8B-In`	**38.3**	**67.5**	**15.6**	**49.2**	7.4	13.5
`aya-expanse-8b`	33.2	67.1	4.8	20.0	**25.6**	**57.5**
`Qwen2.5-7B-In`	29.8	61.5	14.2	46.8	3.31	21.8
One-shot						
`LLaMA-3.1-8B-In`	**45.8**	**72.4**	**35.3**	**67.0**	17.9	53.2
`aya-expanse-8b`	31.7	65.1	29.7	63.7	26.4	59.1
`Qwen2.5-7B-In`	30.2	61.7	18.3	52.8	**35.6**	**60.7**
Fine-tuned						
`LLaMA-3.1-8BIn`	**56.1**	**78.7**	**66.6**	**85.9**	**73.5**	**86.2**
`aya-expanse-8b`	55.0	78.1	62.4	83.7	69.3	86.0
`Qwen2.5-7B-In`	51.9	76.0	63.9	84.1	63.8	78.4

Table 7: MT performance on the COMI-LINGUA test set. B_{en}, B_{rh}, and B_{dh} denote BLEU scores, while cF_{en}, cF_{rh}, and cF_{dh} correspond to chrF++ scores for Standard English, Romanized Hindi, and Devanagari Hindi, respectively. Abbreviations `LLaMA-3.1-8B-In` and `Qwen2.5-7B-In` denote `LLaMA-3.1-8B-Instruct` and `Qwen2.5-7B-Instruct` respectively.

MT, we use the BLEU score (Papineni et al., 2002) and chrF++ score (Popović, 2015) to evaluate the quality of translated outputs. Given the multilingual nature of our dataset, BLEU and chrF++ is computed separately for each output format: B_{en}, cF_{en} for English, B_{rh}, cF_{rh} for Romanized Hindi, and B_{dh}, cF_{dh} for Devanagari Hindi. This disaggregated evaluation helps assess script-specific translation quality and is especially relevant given the transliteration variability in informal code-mixed text.

4.3 Evaluation Configurations

We evaluate model performance under three distinct paradigms: *zero-shot* and *one-shot* in-context learning, and task-specific *fine-tuning*. Traditional NLP tools and libraries are inherently limited to zero-shot settings, as they rely on fixed rule-based or statistical models without the capability for contextual adaptation. In contrast, LLMs are evaluated under both zero-shot, one-shot and fine-tuned configurations to investigate their ability to generalize from instructions alone and to leverage minimal contextual supervision, respectively.

In the zero-shot setting, the prompt includes only task-specific instructions and formatting constraints without any illustrative examples. For the 1-shot setting, we augment the prompt with a single representative example demonstrating the input-output structure of the task. This example is carefully selected to reflect typical task behavior and is kept fixed across all evaluations to maintain consistency. For fine-tuning, we train models on task-specific training splits using formatted instruction-response pairs, allowing models to learn code-mixing patterns and task structures through parameter updates. Detailed illustrations of both prompt configurations for each task are provided in Appendix §B and for fine-tuning, detailed hyperparameters are provided in Appendix §B.1.

5 Results and Observations

Table 4 present the empirical results obtained under the two experimental configurations: zero-shot and one-shot in-context learning, respectively. It is important to note that traditional tools such as `codeswitch` and `Microsoft LID` are limited in their task coverage; results for tasks not supported by these tools are omitted from the tables.

Traditional Tools vs. LLMs: The comparative analysis of traditional NLP tools and LLMs reveals clear distinctions in performance across code-mixed tasks. As shown in Table 4, traditional tools such as `codeswitch` and `Microsoft LID` demonstrate strong performance on specific tasks they were designed for, particularly POS and LID, respectively. For instance, `codeswitch` achieves the highest POS F1 of 88.2, outperforming all LLMs in this task, while `Microsoft LID` attains a reasonable F1 of 74.4 for LID. However, these tools exhibit significant limitations in task coverage; they do not support MLI, MT or tasks involving complex reasoning or generation.

Open vs. Closed LLMs The performance gap between proprietary (closed) and open-weight LLMs is evident across both zero-shot and few-shot settings. In zero-shot mode, closed models such as `gpt-4o` and `claude-3.5-sonnet` dominate with top-tier results in most tasks. For example, `gpt-4o` achieves 92.7 F1 on LID and 98.1 F1 on MLI, while `claude-3.5-sonnet` reaches 92.1 F1 on LID and 90.0 F1 on MLI. However, when moving to a one-shot setting, open-weight models like `LLaMA-3.3-instruct` start closing the gap. Its performance improves significantly: LID F1 rises from 73.3 to 89.3, POS tagging reaches 84.1 (even surpassing `gpt-4o`), and NER climbs to 78.4. MT performance also peaks at 62.2 B_{en} and 74.8 cF_{en} for English, the highest across all models.

Zero vs. One-shot Inference The transition from zero-shot to one-shot inference leads to notable performance improvements across most models and tasks. This is especially evident in complex tasks such as NER and MT, where providing one task-specific instance helps models disambiguate entities and manage code-mixed structures more effectively. For example, `claude-3.5-sonnet`'s NER F1 increases significantly from 56.7 in the zero-shot setting to 85.0 in the one-shot setting, while `LLaMA-3.3-instruct`'s B_{en} improves from 55.4 to 62.2, alongside cF_{en} scores increasing from

71.4 to 74.8 and cF_{rh} from 68.1 to 71.3. `gpt-4o` similarly benefits, with NER performance rising from 60.5 to 77.4 and B_{dh} improving from 32.0 to 58.4 and cF_{dh} from 41.8 to 70.1. Open models like `LLaMA-3.3-instruct` also see considerable gains, such as POS tagging jumping from 68.9 to 84.1 and cF_{en} MT reaching 74.8. These results demonstrate that even minimal supervision through a single example can significantly enhance model performance on linguistically complex, low-resource, or code-mixed tasks. At the same time, tasks like MLI exhibit relatively modest gains, suggesting that more deterministic tasks benefit less from one-shot prompting. Overall, one-shot inference provides a practical and effective method to unlock the latent capabilities of LLMs in multilingual and code-mixed scenarios.

Fine-Tuning LLMs with COMI-LINGUA To further explore model performance beyond zero- and one-shot prompting, we fine-tuned the `LLaMA-3.1-8B-Instruct`, `aya-expanse-8b` and `Qwen2.5-7B-Instruct` models separately on each of the five COMI-LINGUA tasks using the respective training splits. Fine-tuning was carried out using task-specific formatted instructions, allowing the model to internalize both code-mixing patterns and structural nuances. Table 6 present the empirical results obtained under the three experimental configurations: zero-shot and one-shot in-context learning and fine-tuning, respectively. The results demonstrate notable improvements across all tasks, with fine-tuned models outperforming traditional tools, open-weight baselines, and in some cases closed-weight LLMs.

Fine-tuning vs. Prompting Approaches The fine-tuning results from Tables 6 and 7 demonstrate substantial performance gains over both zero-shot and one-shot inferencing approaches. Fine-tuned models consistently outperform traditional tools and achieve competitive or superior results compared to closed-weight LLMs across all tasks. Particularly notable improvements are observed in NER (30-40% gains over prompting approaches) and consistent high performance in MLI (>93% F1 across all fine-tuned models). In the MT task, the model yielded BLEU scores of 56.1 for B_{en}, 66.6 for B_{rh}, and 73.5 for B_{dh}. Correspondingly, the chrF++ scores were 78.7 for cF_{en}, 85.9 for cF_{rh} and 86.2 for cF_{dh} MT.

These results highlight the strength of supervised fine-tuning on high-quality, diverse code-mixed data, as provided by COMI-LINGUA. Unlike

zero- or one-shot setups, which depend heavily on prompt engineering and model prior knowledge, fine-tuning allows the model to generalize deeper linguistic patterns and task-specific strategies.

6 Challenges with Current LLMs

A consistent challenge across all models is the inability to accurately handle English borrowings written in Devanagari script—words like "कोड" and "ओलंपिक" were frequently misclassified as Hindi, reflecting a gap in script-aware language identification. Another prominent issue is sentence truncation; longer code-mixed inputs often lead to incomplete or abruptly cut-off outputs, indicating that many models struggle to preserve context over extended sequences. Models such as `gemini-1.5-flash` and `mistral-instruct` displayed repetitive generation patterns, producing redundant phrases within the same response. These models also occasionally injected subjective explanations into their outputs, despite clear instructions to extract objective information. Several models tended to mirror patterns from the prompt rather than perform actual analysis, indicating shallow understanding. Sentences with high grammatical or script variability posed yet another barrier, where many models, especially `gemini-1.5-flash` and `mistral-instruct`, failed to generate any output at all. Overfitting to examples also emerged as a concern, particularly in one-shot settings; models like `gpt-4o` and `command-a-03-2025` occasionally produced outputs that mimicked example structures rather than responding appropriately to the test input. This over-reliance was particularly evident in tasks such as MLI and LID, where one-shot performance slightly declined. Additionally, models hallucinated non-existent entities, suggesting overgeneralization from minimal supervision. (See Table 10 in the Appendix).

Beyond these limitations, our analysis of smaller-weight models (7–8B parameters) uncovered some failure patterns; for MT task, `Qwen2.5-7B-Instruct` inappropriately provided empty outputs with structured labels such as "Unit 1: English, Unit 2: Romanized Hindi, Unit 3: Devanagari Hindi" instead of providing actual translations. In NER tasks, `aya-expanse-8b` misclassified punctuation marks, tagging "(" as Opening parenthesis and ")" as Closing parenthesis rather than using standard entity categories and labelling as 'X'. POS tagging revealed more hallucina-

tion patterns, with `LLaMA-3.1-8B-Instruct` generating repetitive sequences like "VERB NOUN PROPN NOUN NOUN NOUN NOUN" for multiple instances. More concerning was the tendency of these models to output code snippets instead of task responses —`LLaMA-3.1-8B-Instruct` and `Qwen2.5-7B-Instruct` output Python import statements and function templates, rather than returning direct predictions as:

```
import nltk
from nltk import pos_tag
# Download the required NLTK data
nltk.download('perceptron_tagger')
nltk.download('punkt')
```

Similar code generation patterns emerged across tasks, with the model providing `import re` and `langdetect` modules for the LID task rather than providing the actual labels. Entity hallucination was prevalent in `Qwen2.5-7B-Instruct`, which generated anomalous labels like (Live India) as 'X X X X X X' and inappropriately tagged terms such as "आईपीएल" as HASHTAG entity. MT task suffered from incomplete generation, with outputs abruptly ending mid-sentence, as observed in Romanized Hindi Translation: "Madras Hāikōrt ne dāk vibhāg ko." (See Table 11 in the Appendix). These systematic failures across all three smaller models highlight the importance of robust fine-tuning and careful prompt engineering when deploying compact LLMs for complex multilingual tasks.

7 Conclusion and Future Directions

LLMs often struggle with tasks like POS tagging, NER, and MT in code-mixed Hindi-English due to their lack of exposure to Indian multilingual data. Errors such as mislabeling entities or hallucinating content arise from limited training on structurally complex and script-variable inputs. The COMI-LINGUA dataset addresses these issues by providing high-quality, task-diverse, and richly annotated code-mixed text. Fine-tuning on this dataset enables models to better handle linguistic ambiguity, reduce overfitting, and improve reliability across tasks. Its inclusion of contextual examples and diverse sources—like social media and news—enhances the models' ability to generalize across formal and informal registers, while iterative refinement through active learning ensures sustained performance gains.

Limitations

While this study offers valuable insights into the annotation and processing of Hinglish code-mixed text, several limitations warrant consideration:

1. **Language Pair Specificity:** The findings derived from Hinglish code-mixed data may not generalize to other language pairs (e.g., Spanish-English), given differences in syntactic structure, sociolinguistic norms, and code-switching behavior.

2. **Demographic Bias:** The use of a relatively small and homogeneous group of annotators may introduce demographic bias, potentially limiting the broader applicability and reliability of the acceptability ratings.

3. **Resource Constraints:** Scaling this work to other code-mixed language pairs remains challenging due to the scarcity of high-quality annotated corpora and the limited availability of models capable of robustly handling diverse code-mixing phenomena.

4. **Computational Accessibility:** While fine-tuning shows substantial improvements, computational requirements and the need for substantial training data may limit accessibility for resource-constrained settings.

Ethics Statement

We adhere to established ethical guidelines in the creation of our benchmark dataset and in the evaluation of existing LLMs for Hinglish code-mixed text. Data curation was carried out responsibly, with careful attention to the annotator's well-being, informed consent, and workload management. We ensured that no personally identifiable information (PII) was included in the dataset, thereby maintaining user privacy and confidentiality. To mitigate potential biases, annotation protocols were designed to capture diverse linguistic phenomena and were reviewed iteratively. Our study promotes fairness and inclusivity in multilingual NLP by focusing on underrepresented code-mixed language scenarios. All datasets and models employed in this research are either publicly available or used in accordance with their respective licenses, such as Creative Commons.

Acknowledgments

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Gustavo Aguilar, Sudipta Kar, and Thamar Solorio. 2020. LinCE: A centralized benchmark for linguistic code-switching evaluation. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1803–1813, Marseille, France. European Language Resources Association.

Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Mohd Zeeshan Ansari, Tanvir Ahmad, and Md Arshad Ali. 2019. Cross script hindi english ner corpus from wikipedia. In *International Conference on Intelligent Data Communication Technologies and Internet of Things (ICICI) 2018*, pages 1006–1012. Springer.

Anthropic. 2024. Claude 3.5 sonnet model card addendum. https://www-cdn.anthropic.com/fed9cc193a14b84131812372d8d5857f8f304c52/Model_Card_Claude_3_Addendum.pdf. Addendum to the Claude 3 Model Card.

Kalika Bali, Jatin Sharma, Monojit Choudhury, and Yogarshi Vyas. 2014. "I am borrowing ya mixing ?" an analysis of English-Hindi code mixing in Facebook. In *Proceedings of the First Workshop on Computational Approaches to Code Switching*, pages 116–126, Doha, Qatar. Association for Computational Linguistics.

---

[8]More information about the Project is present here: https://lingo.iitgn.ac.in/codemixing

Rafiya Begum, Kalika Bali, Monojit Choudhury, Koustav Rudra, and Niloy Ganguly. 2016. Functions of code-switching in tweets: An annotation framework and some initial experiments. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1644–1650, Portorož, Slovenia. European Language Resources Association (ELRA).

Rupal Bhargava, Bapiraju Vamsi, and Yashvardhan Sharma. 2016. Named entity recognition for code mixing in indian languages using hybrid approach. *Facilities*, 23(10).

Aditya Bohra, Deepanshu Vijay, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A dataset of Hindi-English code-mixed social media text for hate speech detection. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*, pages 36–41, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Arindam Chatterjee, Chhavi Sharma, Ayush Raj, and Asif Ekbal. 2022. Pacman: Parallel codemixed data generation for pos tagging. In *Proceedings of the 19th International Conference on Natural Language Processing (ICON)*, pages 234–244.

Team Cohere, Arash Ahmadian, Marwan Ahmed, Jay Alammar, Yazeed Alnumay, Sophia Althammer, Arkady Arkhangorodsky, Viraat Aryabumi, Dennis Aumiller, Raphaël Avalos, et al. 2025. Command a: An enterprise-ready large language model. *arXiv preprint arXiv:2504.00698*.

Amitava Das and Björn Gambäck. 2014. Identifying languages at the word level in code-mixed indian social media text. In *Proceedings of the 11th International Conference on Natural Language Processing*, pages 378–387.

Mrinal Dhar, Vaibhav Kumar, and Manish Shrivastava. 2018. Enabling code-mixed translation: Parallel corpus creation and MT augmentation approach. In *Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing*, pages 131–140, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Suman Dowlagar and Radhika Mamidi. 2022. Cmnerone at semeval-2022 task 11: Code-mixed named entity recognition by leveraging multilingual data. In *Proceedings of the 16th International Workshop on Semantic Evaluation (SemEval-2022)*, pages 1556–1561.

Joseph L Fleiss. 1971. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378.

Soumitra Ghosh, Amit Priyankar, Asif Ekbal, and Pushpak Bhattacharyya. 2023. Multitasking of sentiment detection and emotion recognition in code-mixed hinglish data. *Knowledge-Based Systems*, 260:110182.

François Grosjean. 2021. *The Extent of Bilingualism*, page 27–39. Cambridge University Press.

Sunil Gundapu and Radhika Mamidi. 2018. Word level language identification in English Telugu code mixed data. In *Proceedings of the 32nd Pacific Asia Conference on Language, Information and Computation*, Hong Kong. Association for Computational Linguistics.

Rahul Gupta, Vivek Srivastava, and Mayank Singh. 2023. MUTANT: A multi-sentential code-mixed Hinglish dataset. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 744–753, Dubrovnik, Croatia. Association for Computational Linguistics.

Kevin Hallgren. 2012. Computing inter-rater reliability for observational data: An overview and tutorial. *Tutorials in Quantitative Methods for Psychology*, 8:23–34.

Asha Hegde and Shashirekha Lakshmaiah. 2022. MUCS@MixMT: IndicTrans-based machine translation for Hinglish text. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 1131–1135, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Anupam Jamatia, Björn Gambäck, and Amitava Das. 2015. Part-of-speech tagging for code-mixed english-hindi twitter and facebook chat messages. In *Proceedings of the International Conference Recent Advances in Natural Language Processing*, pages 239–248.

Anupam Jamatia, Steve Durairaj Swamy, Björn Gambäck, Amitava Das, and Swapan Debbarma. 2020. Deep learning based sentiment analysis in a code-mixed english-hindi and english-bengali social media corpus. *International journal on artificial intelligence tools*, 29(05):2050014.

AQ Jiang, A Sablayrolles, A Mensch, C Bamford, DS Chaplot, D de las Casas, F Bressand, G Lengyel, G Lample, L Saulnier, et al. 2023. Mistral 7b (2023). *arXiv preprint arXiv:2310.06825*.

Ramchandra Joshi and Raviraj Joshi. 2022. Evaluating input representation for language identification in hindi-english code mixed text. In *ICDSMLA 2020: Proceedings of the 2nd International Conference on Data Science, Machine Learning and Applications*, pages 795–802. Springer.

Kartik Kartik, Sanjana Soni, Anoop Kunchukuttan, Tanmoy Chakraborty, and Md Shad Akhtar. 2024. Synthetic data generation and joint learning for robust code-mixed translation. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 15480–15492.

Simran Khanuja, Sandipan Dandapat, Anirudh Srinivasan, Sunayana Sitaram, and Monojit Choudhury. 2020. GLUECoS: An evaluation benchmark for

code-switched NLP. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3575–3585, Online. Association for Computational Linguistics.

Simran Khanujaa, Sandipan Dandapatb, Sunayana Sitarama, and Monojit Choudhurya. 2020. A new dataset for natural language inference from code-mixed conversations. In *LREC 2020 Workshop Language Resources and Evaluation Conference 11–16 May 2020*, page 9.

Prashant Kodali, Anmol Goel, Monojit Choudhury, Manish Shrivastava, and Ponnurangam Kumaraguru. 2022. SyMCoM - syntactic measure of code mixing a study of English-Hindi code-mixing. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 472–480, Dublin, Ireland. Association for Computational Linguistics.

Ritesh Kumar, Aishwarya N. Reganti, Akshit Bhatia, and Tushar Maheshwari. 2018. Aggression-annotated corpus of Hindi-English code-mixed data. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Anoop Kunchukuttan, Pratik Mehta, and Pushpak Bhattacharyya. 2017. The iit bombay english-hindi parallel corpus. *arXiv preprint arXiv:1710.02855*.

Deepthi Mave, Suraj Maharjan, and Thamar Solorio. 2018. Language identification and analysis of code-switched social media text. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 51–61, Melbourne, Australia. Association for Computational Linguistics.

Giovanni Molina, Fahad AlGhamdi, Mahmoud Ghoneim, Abdelati Hawwari, Nicolas Rey-Villamizar, Mona Diab, and Thamar Solorio. 2016. Overview for the second shared task on language identification in code-switched data. In *Proceedings of the Second Workshop on Computational Approaches to Code Switching*, pages 40–49, Austin, Texas. Association for Computational Linguistics.

Rudra Murthy, Pallab Bhattacharjee, Rahul Sharnagat, Jyotsana Khatri, Diptesh Kanojia, and Pushpak Bhattacharyya. 2022. Hiner: A large hindi named entity recognition dataset. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4467–4476.

Ravindra Nayak and Raviraj Joshi. 2022. L3cube-hingcorpus and hingbert: A code mixed hindi-english dataset and bert language models. In *Proceedings of the WILDRE-6 Workshop within the 13th Language Resources and Evaluation Conference*, pages 7–12.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the tenth workshop on statistical machine translation*, pages 392–395.

Adithya Pratapa, Monojit Choudhury, and Sunayana Sitaram. 2018. Word embeddings for code-mixed language processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3067–3072, Brussels, Belgium. Association for Computational Linguistics.

Ruba Priyadharshini, Bharathi Raja Chakravarthi, Mani Vegupatti, and John P. McCrae. 2020. Named entity recognition for code-mixed indian corpus using meta embedding. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)*, pages 68–72.

Royal Sequiera, Monojit Choudhury, and Kalika Bali. 2015. POS tagging of Hindi-English code mixed text from social media: Some machine learning experiments. In *Proceedings of the 12th International Conference on Natural Language Processing*, pages 237–246, Trivandrum, India. NLP Association of India.

Shashi Shekhar, Dilip Kumar Sharma, and Mirza Mohd. Sufyan Beg. 2020. Language identification framework in code-mixed social media text based on quantum lstm — the word belongs to which language? *Modern Physics Letters B*, 34:2050086.

Rajvee Sheth, Shubh Nisar, Heenaben Prajapati, Himanshu Beniwal, and Mayank Singh. 2024. Commentator: A code-mixed multilingual text annotation framework. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 101–109, Miami, Florida, USA. Association for Computational Linguistics.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018a. Language identification and named entity recognition in Hinglish code mixed tweets. In *Proceedings of ACL 2018, Student Research Workshop*, pages 52–58, Melbourne, Australia. Association for Computational Linguistics.

Kushagra Singh, Indira Sen, and Ponnurangam Kumaraguru. 2018b. A Twitter corpus for Hindi-English code mixed POS tagging. In *Proceedings of the Sixth International Workshop on Natural Language Processing for Social Media*, pages 12–17, Melbourne, Australia. Association for Computational Linguistics.

Vinay Singh, Deepanshu Vijay, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018c. Named entity recognition for Hindi-English code-mixed social media text. In *Proceedings of the Seventh Named Entities Workshop*, pages 27–35, Melbourne, Australia. Association for Computational Linguistics.

Dama Sravani and Radhika Mamidi. 2023. Enhancing code-mixed text generation using synthetic data filtering in neural machine translation. In *Proceedings of the 27th Conference on Computational Natural Language Learning (CoNLL)*, pages 211–220, Singapore. Association for Computational Linguistics.

Abhishek Srivastava, Kalika Bali, and Monojit Choudhury. 2020. Understanding script-mixing: A case study of hindi-english bilingual twitter users. In *Proceedings of the 4th Workshop on Computational Approaches to Code Switching*, pages 36–44.

Vivek Srivastava and Mayank Singh. 2020. PHINC: A parallel Hinglish social media code-mixed corpus for machine translation. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 41–49, Online. Association for Computational Linguistics.

Vivek Srivastava and Mayank Singh. 2021a. Challenges and limitations with the metrics measuring the complexity of code-mixed text. In *Proceedings of the Fifth Workshop on Computational Approaches to Linguistic Code-Switching*, pages 6–14.

Vivek Srivastava and Mayank Singh. 2021b. HinGE: A dataset for generation and evaluation of code-mixed Hinglish text. In *Proceedings of the 2nd Workshop on Evaluation and Comparison of NLP Systems*, pages 200–208, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Vivek Srivastava and Mayank Singh. 2021c. Quality evaluation of the low-resource synthetically generated code-mixed Hinglish text. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 314–319, Aberdeen, Scotland, UK. Association for Computational Linguistics.

Sahil Swami, Ankush Khandelwal, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. A corpus of english-hindi code-mixed tweets for sarcasm detection. *arXiv preprint arXiv:1805.11869*.

Gauri Takawane, Abhishek Phaltankar, Varad Patwardhan, Aryan Patil, Raviraj Joshi, and Mukta S Takalikar. 2023. Language augmentation approach for code-mixed text classification. *Natural Language Processing Journal*, 5:100042.

S Thara and Prabaharan Poornachandran. 2018. Code-mixing: A brief survey. In *2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2382–2388.

Paras Tiwari, Sawan Rai, and C Ravindranath Chowdary. 2024. Large scale annotated dataset for code-mix abusive short noisy text. *Language Resources and Evaluation*, pages 1–28.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. 2023. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*.

PV Veena, M Anand Kumar, and KP Soman. 2018. Character embedding for language identification in hindi-english code-mixed social media text. *Computación y Sistemas*, 22(1):65–74.

Deepanshu Vijay, Aditya Bohra, Vinay Singh, Syed Sarfaraz Akhtar, and Manish Shrivastava. 2018. Corpus creation and emotion prediction for Hindi-English code-mixed social media text. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 128–135, New Orleans, Louisiana, USA. Association for Computational Linguistics.

Yogarshi Vyas, Spandana Gella, Jatin Sharma, Kalika Bali, and Monojit Choudhury. 2014. POS tagging of English-Hindi code-mixed social media content. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 974–979, Doha, Qatar. Association for Computational Linguistics.

A Appendix

A.1 Dataset Sources

For dataset collection, we implemented an article-wise scraping process that extracted high-quality data from diverse sources. **News sources** included NDTV[9], ABP News[10], Zee News[11], News18[12], TV9[13], and Aaj Tak.[14] **Digital platforms** like X (formerly "Twitter")[15] and YouTube[16] provided real-time discussions. **Political channels** from INC, BJP, and AAP were included, along with **official sources** such as *Mann Ki Baat*[17] and *Press Information Bureau (PIB)*[18].

A.2 Examples of Noisy Text Instances in the Scrapped Code-Mixed Data

Table 8 presents examples of challenging text patterns identified during manual annotation, including incomplete variants, ambiguous scripts, cross-article concatenation, and mixed-script forms. These were carefully reviewed and, in some cases,

---

[9]https://ndtv.in/
[10]https://www.abplive.com/
[11]https://zeenews.india.com/hindi
[12]https://hindi.news18.com/
[13]https://www.tv9.com/
[14]https://www.aajtak.in/
[15]https://x.com/
[16]https://www.youtube.com/
[17]https://www.narendramodi.in/mann-ki-baat
[18]https://pib.gov.in/

removed as part of our annotation methodology and quality assurance process to improve dataset consistency.

A.3 Annotation Guidelines for All Tasks

- Each instance was annotated independently by all annotators without influence from model predictions or other annotators' decisions.

- Annotators were instructed to rely on contextual understanding to disambiguate code-mixed tokens, resolve ambiguity, and accurately assign labels.

- Only the content explicitly present in the sentence was to be annotated; annotators were advised to avoid adding any inferred or assumed information.

- Instances containing noise (e.g., incomplete fragments, junk tokens, or malformed words) were marked and excluded during preprocessing as per filtering heuristics as per Table 8.

- Consistent labeling was promoted using uniform tags and task-specific instructions during annotation training.

- Annotators were encouraged to flag uncertain and ambiguous samples for further review.

- Annotation disagreements were addressed using majority voting. In cases where no majority existed, a manual adjudication process was conducted to finalize the labels.

Quality Control & Training

- Annotators periodically used gold-standard examples to ensure continued alignment throughout the annotation process.

- Periodic sample checks provided feedback and helped uphold annotation standards.

- An independent reviewer regularly flagged low-quality annotations for re-annotation by the original annotators.

Conflict Resolution Strategy

- Consolidated annotation criteria: For model training and evaluation, only annotations with agreement from at least two out of three annotators were retained, ensuring reliability.

- Iterative refinement: Disagreement patterns were analyzed to identify common sources of confusion, leading to guideline refinements and additional training for annotators.

B Experimental Setup

Zero-shot LID Prompt

You are an expert in the Language Identification (LID) task for Hinglish code-mixed text. Your task is to identify and classify tokens in the given sentence.
Instructions:
- Tag each word or word group in the following text with language labels:
- Use 'hi' for Hindi words
- Use 'en' for English words
- Use 'ot' for other words
- Only break tokens at spaces.
Process the given sentence:
Input: {text}
Return the output in the following format:
$[word1\ tag1,\ word2\ tag2, \ldots]$

One-shot LID Prompt

You are an expert in the Language Identification (LID) task for Hinglish code-mixed text. Your task is to identify and classify tokens in the given sentence.

Instructions:
- Tag each word or word group in the following text with language labels:
- Use 'hi' for Hindi words (e.g., Mujhe, गुरुवार , karna, सुविधा, hai, निर्माण, shala)
- Use 'en' for English words (e.g., Culture, ऑफिस, Lifestyle, for, इंडिया, Alliance, इंटरनेशनल, of, initiative, हेरिटेज)
- Use 'ot' for other words (e.g., #Bollywood, #BJP, @PMOIndia, . - : @ = & * +)
- Be precise and consistent with tags classification.
- Do not add any other extra suggestions.
- Only break tokens at spaces.
- Format: Return space-separated word-tag pairs.
Example Input: मीराबाई चानू ने 21st Commonwealth Games में India के लिए first Gold medal जीता था ।
Output: मीराबाई hi चानू hi ने hi 21st ot Commonwealth en Games en में hi India en के hi लिए hi first en Gold en medal en जीता hi था hi । ot
Input: {text}

Zero-shot MLI Prompt

You are a helpful AI Assistant and your task is to identify and determine the **matrix language (dominant grammatical structure language)** of the given Hinglish (Hindi-English) code-mixed sentence.

Instructions:
The matrix language is the main language that governs the grammatical structure of the sentence. It may borrow words from another language, but the syntax and morphology will mostly follow the matrix language.
Process the given sentence:
Input: {text}
Return the output in the following format: only the matrix language name.

One-shot MLI Prompt

Your task is to identify and determine the matrix language (dominant grammatical structure language) of the given Hinglish (Hindi-English) code-mixed sentence.

The matrix language is the main language that governs the grammatical structure of the sentence. It may borrow words from another language, but the syntax and morphology will mostly follow the matrix language.

Instructions:
1. If the sentence is primarily structured in Hindi, respond with: 'hi'
2. If the sentence is primarily structured in English, respond with: 'en'
3. Respond with a single word only: 'hi' or 'en'. Do not add any other extra suggestions.

Example Input: India's automation and design expert pool is vast, और ज़्यादातर Global कंपनियों के इंजीनियरिंग सेंटर भी भारत में हैं |
Output: en

Process the given sentence:
Input: {text}

Zero-shot NER Prompt

You are a helpful AI Assistant, and your task is to identify the named entities in the following Hinglish code-mixed sentence.
Instructions:
- Tag each word with one of these entity types:
PERSON, ORGANISATION, LOCATION, DATE, TIME, GPE, HASHTAG, EMOJI, MENTION, X - for all other words.

Process the given sentence:
Input: {text}
Return the output in the following format:
[{ 'word1': 'entity'}, { 'word2': 'entity' }, { 'word3': 'entity' }, . . .]

One-shot NER Prompt

You are a helpful AI Assistant, and your task is to identify the named entities in the following Hinglish code-mixed sentence.

Instructions:
- Tag each word with one of these entity types:
PERSON - for names of people
ORGANISATION - for organization names
LOCATION - for location names
DATE - for dates
TIME - for time expressions
GPE - for geo-political entities
HASHTAG - for words starting with #
EMOJI - for emojis
MENTION - for words starting with @
X - for words that don't fall into the above categories.
- Only break tokens at spaces.
- Do not add any extra explanations or text before or after the list.
Example: लंदन के Madame Tussauds में Deepika Padukone के वैक्स स्टेच्यू का गुरुवार को अनावरण हुआ |
Output: [लंदन GPE, के X, Madame LOCATION, Tussauds LOCATION, में X, Deepika PERSON, Padukone PERSON, के X, वैक्स X, स्टेच्यू X, का X, गुरुवार DATE, को X, अनावरण X, हुआ X, | X]
Input: {text}

7987

Zero-shot POS Tagging Prompt

Your task is to assign Part-of-Speech (POS) tags to each word or word group in the given code-mixed sentence.

Instructions:
- Tag each word with the appropriate grammatical category from the provided tagset.
- Use available POS tags: VERB, NOUN, PRON, ADJ, ADV, ADP, PROPN, CONJ, DET, NUM, PART, PRON_WH, PART_NEG, X.
- Only break tokens at spaces.

Process the given sentence:
Input: {text}
Return the output in the following format:
[[{'word1':   'POS_TAG1',   'word2': 'POS_TAG2'}, {'word3': 'POS_TAG3', 'word4': 'POS_TAG4'}, ...]]

- ADV: Adverbs
- ADP: Adpositions (pre/postpositions)
- PRON: Pronouns
- DET: Determiners
- CONJ: Conjunctions
- PART: Particles
- PRON_WH: Question words
- PART_NEG: Negation words
- NUM: Numbers
- X: Other (punctuation, foreign words)

Example Input: मीराबाई चानू ने 21st Commonwealth Games में India के लिए first Gold medal जीता था ।
Output: ['मीराबाई': 'PROPN', 'चानू': 'PROPN', 'ने': 'PART', '21st': 'NUM', 'Commonwealth': 'PROPN', 'Games': 'PROPN', 'में': 'ADP', 'India': 'PROPN', 'के': 'ADP', 'लिए': 'ADP', 'first': 'ADJ', 'Gold': 'NOUN', 'medal': 'NOUN', 'जीता': 'VERB', 'था': 'VERB', '।': 'X']
Input: {text}

One-shot POS Tagging Prompt

You are a linguistics expert specializing in Part-of-Speech (POS) tagging, particularly for code-mixed Hindi-English (Hinglish) text.
Given a Hinglish sentence, provide a token-wise POS tag for each word in JSON format. Ensure accurate tagging for both Hindi and English words, considering the context and mixed grammar structures.

Instructions:
1. Analyze each word in the sentence and identify the correct POS tag.
2. Be precise and consistent with POS classification.
3. Consider the grammatical context of code-mixed structures.
4. Do not add any other extra suggestions.
5. Use only the following tagset:
- NOUN: Common nouns
- PROPN: Proper nouns
- VERB: Verbs in all forms
- ADJ: Adjectives

Zero-shot MT Prompt

You are a helpful AI Assistant specializing in machine translation for code-mixed Hindi-English (Hinglish) text. Your task is to translate Hinglish sentences into three different formats while maintaining meaning and natural flow.

Given a Hinglish code-mixed sentence, provide translations in the following three formats:
1. **Standard English**: Complete fluent and grammatically correct English translation.
2. **Romanized Hindi**: Complete translation in Hindi using Roman/Latin script.
3. **Devanagari Hindi**: Complete translation in fluent Hindi using Devanagari script.

Process the given sentence:
Input: {text}
Return the output in the following format:
Standard English: [English translation]
Romanized Hindi: [Hindi in Roman script]
Devanagari Hindi: [Hindi in Devanagari script]

One-shot MT Prompt

> You are a helpful AI Assistant specializing in machine translation for code-mixed Hindi-English (Hinglish) text. Your task is to translate Hinglish sentences into three different formats while maintaining meaning and natural flow.
>
> Given a Hinglish code-mixed sentence, provide translations in the following three formats:
>
> 1. **Standard English**: Complete fluent and grammatically correct English translation.
> 2. **Romanized Hindi**: Complete translation in Hindi using Roman/Latin script.
> 3. **Devanagari Hindi**: Complete translation in fluent Hindi using Devanagari script.
>
> **Instructions:**
>
> 1. Ensure all translations convey the same meaning as the original Hinglish text.
> 2. Maintain natural flow and grammatical correctness in each target format.
> 3. Consider cultural context and idiomatic expressions appropriately.
> 4. Do not add any other extra suggestions or explanations.
>
> **Example Input:** दिल्ली स्थित Indian Navy मुख्यालय और वेस्टर्न नेवल कमांड ने मिलकर ऑपरेशन के व्यूह की रचना की है ।
>
> **Output:**
>
> English Translation: The Indian Navy headquarters located in Delhi and the Western Naval Command have jointly orchestrated the formation of the operation.
>
> Romanized Hindi Translation: Dilli sthit Indian Navy headquarters aur Western Naval Command ne milkar operation ke vyuh ki rachna ki hai .
>
> Devanagari Hindi Translation: दिल्ली स्थित भारतीय नौसेना मुख्यालय और वेस्टर्न नेवल कमांड ने मिलकर ऑपरेशन के व्यूह की रचना की है ।
>
> Process the given sentence:
>
> **Input:** {text}

B.1 Fine-Tuning Hyperparameters

The optimization process focused on fine-tuning four key hyperparameters, with the goal of balancing the refinement of essential parameters while minimizing unnecessary adjustments to those already well-suited for the task. Batch size, number of epochs, weight decay, and learning rate were selected due to their direct and substantial impact on model performance, stability, and generalization.

Core Training Parameters:

Epochs: 3

Batch size: 4 per device with gradient accumulation steps of 8 (effective batch size: 32)

Learning rate: 2e-4 with cosine scheduler and warmup ratio of 0.1

Weight decay: 0.01

LoRA Configuration: rank 32, alpha 64 and dropout 0.1.

Instruction Format: All tasks used task-specific instruction templates with examples, following the format:

```
Instruction: [task description]
Sentence: [input]
Output: [expected output]
```

C Computation Requirement and Budget

The experiments were conducted using API-based access to state-of-the-art Large Language Models (LLMs), including `gpt-4o`, Command R+ (`command-a-03-2025`) by Cohere, and `claude-3.5-sonnet`. The estimated monthly costs for API usage were approximately $200 for `claude-3.5-sonnet`, $150 for Cohere, and $50 for `gpt-4o`, resulting in a total estimated cost of **$400** per month. For computational infrastructure, experiments were carried out on four NVIDIA Tesla V100 32 GB GPUs, with an estimated cost of $7,192.00 per month based on Google Cloud Platform (GCP) [19] Calculator pricing.

---

[19] https://cloud.google.com/products/calculator

Category	Example Text
Incomplete variant	), floppy disk, hard disk drive, magnetic stripe card, relational database, SQL जीता (DRAM) (Dynamic Random-Access Memory) था
Ambiguous script	Menu प्रोग्रामिंग भाषा .jpg|thumb]] ===++ Image शामिल / [[:en:Giridhar Lal Aggarwal Freedom Fighter | Giridhar Lal Aggar-wal]] == |
Cross-article concatenation	[[चित्र:गिरिधर लाल अग्रवाल [.....] 08/10/2020 Satyam KushwahLeave a Comment on श्री गिरिधर लाल अग्रवाल |
Mixed-script variant	@Strawberigloz he barobar naahi aahe, aaplich manasa aaplyala paathi sodtat. Aaplya itithasacha garva asla pahije.

Table 8: Examples of noisy text instances in the dataset containing mixed content and transitions. *Takeaway*: *These noisy text instances in the dataset reflect challenges in code-mixed annotation and require careful preprocessing.*

Task	Data Source (Hi-En)	Dataset Size	Script	QA	Annotators/Models
LID	Facebook (Bali et al., 2014)	1,062	R & D	Yes	3
	Twitter (Singh et al., 2018a)	2,079	R	Yes	3
	Twitter (Swami et al., 2018)	5,250	R	Yes	Not mentioned
	Twitter (Mave et al., 2018)	5,567	R	Yes	3
	Facebook, Twitter, WhatsApp (Veena et al., 2018)	3,071	R	No	Embedding Model
	Twitter (Joshi and Joshi, 2022)	18,461	R	No	Not mentioned
	Twitter, YouTube, Press Releases, News (Ours)	**25,773**	**R & D**	**Yes**	**3**
MLI	Twitter, Facebook (Sequiera et al., 2015)	628	R & D	No	1
	Facebook (Bali et al., 2014)	1,062	R & D	Yes	3
	Social Media (Dhar et al., 2018)	6,096	R	Yes	4
	Twitter, YouTube, Press Releases, News (Ours)	**25,773**	**R & D**	**Yes**	**3**
NER	Facebook (Bali et al., 2014)	1,062	R & D	Yes	3
	Twitter (Singh et al., 2018a)	2,079	R	Yes	3
	Twitter (Bhargava et al., 2016)	2,700	R	No	Supervised algorithm
	Twitter (Singh et al., 2018c)	3,638	R	Yes	2
	Tourism, News (Murthy et al., 2022)	108,608	R & D	No	1
	Twitter, YouTube, Press Releases, News (Ours)	**24,913**	**R & D**	**Yes**	**3**
POS	Twitter, Facebook (Sequiera et al., 2015)	628	R & D	No	1
	Facebook (Bali et al., 2014)	1,062	R & D	Yes	3
	Twitter, Facebook (Jamatia et al., 2015)	1,106	R	No	2
	Twitter (Singh et al., 2018b)	1,190	R	Yes	3
	Synthetically generated (Chatterjee et al., 2022)	51,118	R & D	No	0
	Existing Benchmarks (Kodali et al., 2022)	55,474	R	No	Trained POS tagger
	Twitter, YouTube, Press Releases, News (Ours)	**24,598**	**R & D**	**Yes**	**3**
MT	TED Talks, News, Wikipedia (Kartik et al., 2024)	2,787	R & D	Yes	2
	Twitter, Facebook (Srivastava and Singh, 2021b)	3,952	R & D	Yes	5
	Social Media (Dhar et al., 2018)	6,096	R	Yes	4
	Twitter, Facebook (Srivastava and Singh, 2020)	13,738	R	Yes	54 (400 instances)
	Existing Benchmarks (Kunchukuttan et al., 2017)	14,95,854	R & D	No	PBSMT, NMT
	Twitter, YouTube, Press Releases, News (Ours)	**24,558**	**R & D**	**Yes**	**2**

Table 9: Comprehensive Comparison of Existing Datasets for Hinglish Code-Mixing NLP Tasks, including the proposed dataset. NLP tasks covered in the dataset include Language Identification (LID), Part-of-speech (POS) tagging, Named Entity Recognition (NER), Matrix Language Identification (MLI) and Machine Translation (MT). (R) and (D) denote Roman and Devanagari scripts, respectively, while QA represents annotations by Qualified Annotators. *Takeaway: The COMI-LINGUA dataset comprises 1,25,615 manually annotated instances by three annotators across five NLP tasks.*

Response Flaw Type	Example Behavior or Observation
Script and entity Misidentification	Words such as 'लंदन', which are borrowed English terms written in Devanagari, are frequently misclassified as Hindi by most models. Additionally, models like `gpt-4o` demonstrate entity misclassification issues, such as tagging 'Union Home Minister' as an ORGANISATION and 'Holi' as a DATE.
Sentence Truncation	Long-form code-mixed inputs lead to abrupt endings or incomplete generations (e.g., output stops mid-sentence despite ample context). In 'Yes, we belong to this place – ये भाव आज हम अपने युवाओं में देख रहे हैं। ', only 'Yes, we belong to this place' is translated to (हाँ, हम इस जगह से संबंधित हैं / haan, hum is jagah se sambandhit hain), while the rest is ignored despite clear context.
Repetitive Generation	Models like `gemini-1.5-flash` and `mistral-instruct` frequently exhibit repetitive generation patterns. For instance, they may produce outputs such as: 'The second tagging is more accurate as it identifies 'this' as a determiner and 'last' as a quantity'. repeating similar explanations or sentence fragments within the same response.
Subjective Additions	Instead of remaining factual, models add speculative commentary (e.g., 'en: The given text is in English. The hashtag '#MadeByGoogle' is also in English. 'E' (English).').
Prompt Mimicry	`gpt-4o` and `command-r-plus` mirror example formats from the prompt, failing to adapt to new inputs and instead mimicking example structure. 'Based on the given text, it is written in the Hindi language. Therefore, the matrix language label for this sentence is h'.
High-variance Failure	Inputs with abrupt transitions, broken grammar, or inconsistent scripts result in empty, irrelevant, or default responses. **Example 1:** lakhanOo: dr. apj abdul kalam bhArat kA 11veN rAShTrapati thE karoDhON bhAratiyON ke lIyE prEraNAdhA kA strOt thE, dR. apj abdul kalam **Example 2:** This text does not contain any GPE, DATE, TIME, HASHTAG, EMOJI, or MENTION entities.
Hallucination	Models like `mistral-instruct` fabricate non-existent locations or attributes (e.g., inventing MATCH, VERSION, COUNTRY, PRODUCT, QUANTITY, or BUILDING categories not present in the input). **Example 1:** Note: The context is assumed to be empty in this example. If context information is available, it should be provided to improve the accuracy of entity tagging. **Example 2:** ḍiptī sīṭī mānejar lēmūēla rāṇḍolpha nē kahā, 'splāś pāḍ kā rōjānā sāf-sāfāī kamī huī hai

Table 10: Observed limitations across LLMs while processing noisy, code-mixed text. ***Takeaway***: *Failures are diverse - ranging from linguistic issues to structural hallucinations and prompt sensitivity - highlighting the need for integrated data-centric training strategies that can effectively handle linguistic and structural complexities.*

Response Flaw Type	Example Behavior or Observation
Output Blank or Missing	The `Qwen2.5-7B-Instruct` and `LLaMA-3.1-8B-Instruct` models frequently returned blank or no output for complex sentences across all tasks, particularly in zero-shot settings when encountering noisy or script-mixed inputs.
Task Instruction Violation	Models like `aya-expanse-8b` and `LLaMA-3.1-8B-Instruct` generated Python code snippets in some task outputs: '# Split the input text into words based on spaces `words = text.split()` # Initialize an empty list for words `tagged_words = []` # Iterate over each word in input text given `for word in words:` # Check if the word is in the Hindi' for LID tasks, failing to follow prompt formatting requirements and giving incorrect output.
Subjective Commentary Injection	All Models added unnecessary disclaimers like 'Please note that the output is a best-effort attempt and might not be 100% accurate due to the complexity of Hinglish language' and 'Please let me know if you need any further assistance!' instead of providing direct outputs.
Language Script Confusion	`aya-expanse-8b` exhibited severe input-output disconnection and incorrect script identification in MT and MLI tasks. For MT, an input about Bangladesh football team yielded an unrelated output discussing Republic TV and Arnab Goswami in "Urdu" script. For MLI, the input 'इस बीच, बांग्लादेश फुटबॉल टीम...' was misclassified as "Bengali (Bangla)" instead of Hindi, indicating a failure to correctly identify the matrix language.
Entity Type Misclassification	All Models frequently misclassified entities in code-mixed contexts, such as tagging 'Uttar Pradesh' as separate mismatched entities ('Uttar': ORGANISATION, 'Pradesh': PLACE) instead of the correct unified LOCATION labels ('Uttar': LOCATION, 'Pradesh': LOCATION), demonstrating poor understanding of entity boundaries.
Hallucination	`aya-expanse-8b` in one-shot POS tagging created its own input sentence ('वहाँ पर मैंने एक बहुत ही सुंदर दृश्य देखा') different from the actual input and monolingual, then tagged the fabricated sentence.
Inconsistent Output Formats	`LLaMA-3.1-8B-Instruct` and `Qwen2.5-7B-Instruct` provided inconsistent MLI labels like 'Mixed', 'Code-mixed language: Hindi-English' instead of 'hi' or 'en', for some inputs, showing format instability across evaluation instances.
Multi-language Script Errors	For zero-shot MT, `Qwen2.5-7B-Instruct` generated Romanized Hindi in Arabic-Urdu and Bangla-English script instead of Latin script, completely misunderstanding and hallucinating in some input instances.

Table 11: Observed limitations across 7–8B parameter LLMs during zero-and one-shot evaluation. ***Takeaway***: *While smaller models exhibit severe failure patterns in zero-shot and one-shot settings, fine-tuning on code-mixed data transforms them into highly capable systems that often match or exceed larger proprietary models, demonstrating the importance of task-specific training for deploying compact models in multilingual scenarios.*