Do LLMs Model Human Linguistic Variation? A Case Study in Hindi-English Verb Code-Mixing

Anonymous ACL submission

Abstract

Can large language models (LLMs) model linguistic variation? We investigate this question through Hindi-English (Hinglish) verb code-mixing, where speakers can use either a Hindi verb or an English verb with the light verb *karna* ('do'). Both forms are grammatical, but speakers show unexplained variation in language choice for the verb. We compare human preferences on controlled code-mixed minimal pairs to LLM perplexities spanning families, sizes, and training language compositions. We find that current LLMs cannot reliably classify verb language preferences to match native speaker judgments. We also see that with specific supervision, some models can predict human preference to an extent. We release native speaker acceptability judgments on 30 verb pairs, perplexity ratios for 4,279 verb pairs across 7 models, and experimental materials.

1 Introduction

> *But the point isn't whether a text produced by any system contains words from more than one language - it is rather whether they're mixed in the way that a bilingual human might.* (Sterner and Teufel, 2025)

Social Science and policy researchers are increasingly using LLMs as human behavioral proxies: to simulate American survey responses on new surveys (Hewitt et al., 2024), qualitative interviewing of replicas fine-tuned on thousands of real demographic backstories (Argyle et al., 2023), and more. However, parallel literature also finds that LLMs are not grounded in human motives or embodiment, so in scenarios requiring strategic thinking, genuine uncertainty (Gao et al., 2025), or deeply subjective, story-driven research (Kapania et al., 2024), LLMs cannot be a reliable proxy for human psychology.

Meanwhile, linguistics research shows us that LLMs are already good at making human-like decisions on grammaticality judgments across languages (Jumelet et al., 2025) but struggle at matching human linguistic intuition in novel contexts like suffixing nonce adjectives (Weissweiler et al., 2025). These debates prompt us to think that while modeling human social behavior is a complex task because of its high subjectivity, to what extent can LLMs model human *linguistic variation*?

We test this question on the phenomenon of Hindi–English (Hinglish) verb code-mixing: As shown in Fig.1, for a given Hindi predicate, native speakers can use either a Hindi verb or an English verbal noun followed by the Hindi verbalizer *karna* ('to do'). We find this phenomenon a neat test of LLM-human linguistic variation alignment as: (a) Replacing the Hindi verb in a sentence with its Hinglish counterpart does not make it ungrammatical, (b) however native speakers find some Hinglish verbs awkward to mix in, some comfortable, and some where they're preferred over the Hindi verb. (c) Finally, the phenomenon has not yet been quantified or theorized about.

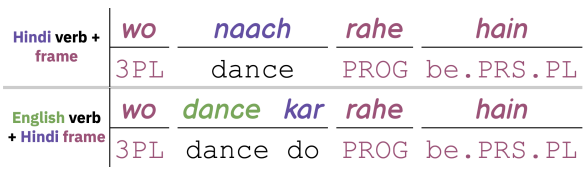


Figure 1: Hindi vs. Hinglish code-mixed at verb, for: *They*(3PL) + *are*(be.PRS.PL) + *danc-ing*(dance-PROG).

We design an experiment to collect human preferences over 30 verbs of varying degrees of mixability in context of sentence pairs (Fig.1). We then compare them to LLM perplexity ratios of the same sentence pair. Our findings show that LLMs –across, sizes, families, and degrees of Hindi-English bilingual training data– cannot model human linguistic variation shown by this phenomenon. Further experimentation with careful feature selection and supervision showed the highest (but insufficient) alignment by using Sarvam-1 (AI, 2024) (F1=0.81). Fur-

ther, this performance is restricted to one model and doesn't generalize across models under the same supervised and feature selected conditions.

Apart from the broader findings on LLM-usability for linguistic variation studies, this work also provide the first systematic evaluation of human Hinglish verb code-mixing preferences. We release native bilingual preference ratings focused on this phenomenon for 30 verb pairs, perplexity ratios for 4,279 verb pairs across 7 base LLMs, and the data.

2 Background

2.1 Code-mixing and the *Do-support* verb in Hinglish

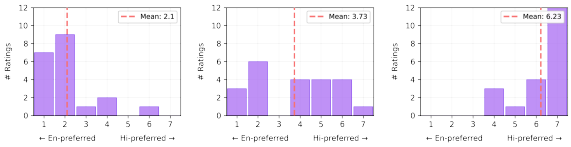
Code-mixing is a term used to describe the process of native speakers embedding words or phrases from one language (*embedded language*) into the grammatical structure of another (*matrix language*) (Myers-Scotton, 1997).

In Hindi-English code-mixing, Hindi is typically the *matrix language*, and mixing is predominantly *insertional*, i.e. English items are inserted into Hindi syntactic frames. So, English words inflect or combine in ways (e.g., taking Hindi postpositions) to maintain grammaticality in Hindi. These frameworks account for the grammatical *shape* of mixing, but also leave room for *choice*, i.e. which words get mixed –and how often– varies across items and speakers (Bali et al., 2014).

In Hinglish, predicates are majorly expressed in two ways: either the native Hindi verb, or an English noun/adjective followed by the Hindi light verb *karna* (do) (Dey and Fung, 2014). Both versions are grammatical as the pattern of mixing aligns with how Hindi compound verbs work independently: the first element contributes core lexical meaning, while the light verb (e.g., *karna*) supplies aspect/tense and other functional morphology (Kumar, 1986). However, as shown by Fig.1 (a commonly spoken mix) and Fig.2 (an unacceptable mix), *not all Hinglish mixes are equally natural to native speakers*.

Hindi verb + frame	*khaana*	*aasaan*	*hai*
	eat.INF	easy	be.PRS.SG
English verb + Hindi frame	*eat karna*	*aasaan*	*hai*
	eat do.INF	easy	be.PRS.SG

Figure 2: Unnatural but grammatical, for: (*It*(EXPL) + *is*(be.PRS.PL)) + *easy*(easy) + *to*(INF) + *eat*(eat).



(a) kiss - चूम (b) change - बदल (c) tear - फाड़

Figure 3: Human preference histograms. x-axis: En-preferred to Hi-preferred (1-7), y-axis: # ratings (0-12) while all the verbs are fairly frequent, *kiss* is preferred at Fig.3a, फाड़ is preferred at Fig.3c, while Fig.3b shows easy mixability

2.2 Acceptability Variation and Why its Difficult to Model

The variation in naturalness of the code-mixing acceptability thus lies on a *continuum*. Bali et al. (2014)'s frequency based corpus analysis shows: *extreme conditions* - highly frequent English items that are readily mixed vs. near-absent ones that speakers avoid, and a *large middle* where both Hindi and English variants are used with comparable rates.

However as 3 shows, in the case of verbs, even with fairly frequent verb-pairs in Hinglish, we observe all 3 parts of the continuum. There is no theory that explains Hinglish verb mixing variation fully, yet. This could happen due to various reasons in addition to lexical frequency: *Domain dominance* (English forms are preferred in tech, work, pop culture), *No native substitute / lexical gap* (English is also chosen for proper names, terms of art), *Degree of bilingualism* (Dey and Fung, 2014), *Attitudes toward mixing* (positive attitudes imply higher acceptability (Badiola et al., 2018)).

Recent empirical work by Kodali et al. (2025) shows that structure-only metrics of "how mixed" a sentence is: such as the *Code-Mixing Index (CMI* (Gambäck and Das, 2016)*)* and simple *number-of-switch* counts (Pratapa et al., 2018), correlate poorly with human acceptability for Hinglish. In contrast, fine-tuned multilingual language models trained on human judgments substantially outperform such metrics at distinguishing natural vs. awkward code-mixed sentences. This indicates that language models can model general Hinglish acceptability to a certain extent.

Our focus: Building on this, we specifically study the verb code-mixing phenomenon in Hinglish. As it presents an isolated, interpretable, test bed for LLM alignment to human linguistic variation in verb language choice.

3 Data

Preference data was prepared by constructing Hindi **frames** (Sec.3.2), scraping Hindi-English **verb pairs** to insert in the frames (Sec.3.1), and present them in different orthographies/**forms** (Sec.3.3).

3.1 Verb Pairs

The verb pairs for the experiment were obtained from Indowordnet (Bhattacharyya, 2010) parallel lexicon. The lexicon was pre-processed (detailed at App.C) to retain verbs, remove phrasal verbs, etc. to obtain a total of 4,279 unique verb pairs.

3.2 Frames

We created a total of 5 Hindi frames (Tab.1) to control for:

1. **Length**: Frames were limited to 6 words (∼12-15 tokens) as LLM perplexity scores become less reliable and interpretable for longer sentences (Hu et al., 2024).
2. **Grammaticality**: Frames were grammatically compatible with most Hindi verbs *and* their English counterparts with 'do' support.
3. **Minimal confounds**: Main verb was the only content word, non-verbal confounders were removed, ensuring maximum of one subject (third person, plural (3PL) pronoun, in the nominative/agent case).[1]
4. **Verbal confounds**: Frames contain variants of verbs positions (beginning, middle, end), adverbs (positive and negative senses), and grammatical features like tense, and aspect [2].

3.3 Forms

We consider three orthographic forms for presenting Hindi frames and verbs:

(a) **Devanagari**: Native script (e.g., वो नाच रहे हैं)
(b) **Casual Latin**: Native speaker romanization in free variation (e.g., *wo naach rahe hain*)
(c) **Formal Latin**: Standardized rule-based transliteration (Gupta et al., 2010) (e.g., *vo naacha rahe hain*)

These cover the majority of Hinglish content online and thus, LLM training data (Kodali et al., 2025). For our experiments, we transliterated Devanagari using GPT-4.1 with iteratively refined prompts by native speaker evaluation. Complete prompts, development processes, and statistical analyses are provided in App.A.

The resultant five frames for both the Hindi and English verb counterparts in all three transliterations are presented in Tab.7.

4 Experimental Design

We conducted experiments on humans and LLMs to collect preferences on the data above to check if LLMs can predict human linguistic variation.

4.1 LLM Preferences

We choose 7 open-weights LLMs: Gemma3-1B, Gemma3-4B, Gemma3-12B (Team et al., 2025), Llama3.1-8B, Llama3.1-70B (Grattafiori et al., 2024), Qwen3-32B (Yang et al., 2025), and Sarvam-1 (2B)[3]. These models were chosen to span a range of popular architectures and sizes, covering multiple model families and training language mixes. We included different model sizes within the same family (e.g., the Llama3.1 series) to examine scaling effects, and we used each model's base (non-instruct) version to avoid preference- and instruct-tuning biases. Notably, aside from Sarvam-1, none of these models had been trained from scratch on substantial mixtures of Indic languages (base checkpoints for such models are generally unavailable).[4]

We quantify an LLM's preference for a sentence using perplexity. For a sentence $x_{1:N}$ of length N, the perplexity under model θ is defined as

$$\text{ppl}_\theta(x_{1:N}) = \exp\left(-\frac{1}{N}\sum_{i=1}^{N}\log p_\theta(x_i \mid x_{<i})\right) \tag{1}$$

which corresponds to the model's average per-token surprise (Eqn.1). To compare a Hindi sentence with its English counterpart, we use the perplexity ratio as our metric. For a given Hindi sentence x_{hi} (in any of the three forms) and its English version x_{en}, we define the following ratio (Eqn.2)

$$PR_\theta(x_{\text{hi}}, x_{\text{en}}) = \frac{\text{ppl}_\theta(x_{\text{hi}})}{\text{ppl}_\theta(x_{\text{en}})} \tag{2}$$

[1] in Hindi, the 3PL form (*wo*) does not need differential verb inflection and works with all nouns.

[2] Modality variations would have made frame lengths more disparate in Hindi as they are represented by separate auxiliary particles in the verb phrase

[3] The data collection was run using Hugging Face Transformers (Wolf et al., 2020) with BF16 quantization, on 2x NVIDIA RTX 6000 GPUs running for 10 GPU hours. Refer App.D for more details

[4] We also considered closed-source models, but their APIs did not support extracting logprobs for sequences, full probability distributions or token-level cross-entropies, rendering them unsuitable for our perplexity-based analysis.

Frames	Gloss	Meaning
wo X (kar) rahe hain	3PL/SG X (do) PROG be.PRS.PL	'They are X-ing'
X(kar)na aasan hai	X (do.)INF easy be.PRS.SG	'Doing X is easy'
wo kal X (kar) chuke the	3PL/SG yesterday X (do) finish.PERF be.PST.PL	'They had finished doing X yesterday'
X(kar)na mana hona chahiye	X (do.)INF forbidden be.INF should	'Doing X should be forbidden'
wo aaram se X(kar)te rahenge	3PL/SG ease with X (do.)HAB continue.FUT.PL	'They will keep doing X'

Table 1: Hindi (and English) frame templates with glosses and translations. X marks the spot of the verb.

This ratio serves as the unit of comparison between models and humans: a ratio below 1 indicates the model assigns lower perplexity (higher preference) to the Hindi sentence, whereas a ratio above 1 indicates a preference for the English sentence.

We make these perplexities and ratios over all the verb pairs in their 5 frames x 3 forms public on our repository as well.

4.2 Human Preferences

4.2.1 Participants

Participants were recruited via Prolific (2025), inclusion criteria required participants to be: 18-65 years old, native Hindi speakers, and fluent in English. We collected responses from 18 participants, removed an outlier on the basis of z-score analysis on responses, and 3 others due to incomplete preference data, yielding **14** for the final analysis.[5]

4.2.2 Data Curation

We sampled 30 verb pairs and selected 1 form, as collecting human judgments for all 4,279 verb pairs across 15 conditions (5 frames x 3 forms) would be expensive (Sterner and Teufel, 2025):

Verb Pairs: We sample 30 verb pairs for the Human experiment using a hybrid strategy. First, we use k-means clustering using 18 features for each pair: Llama3.1-70B[6] PR_θ (from Eqn.2) for all 15 combinations (5 frames x 3 forms) + min, max, and std. deviation along the 15 features. This results in 4 clusters from which we randomly sample 100 pairs (25 per cluster) ensuring diversity and coverage of the verb space in terms of LLM preference.

From these 100, two native speaker judges selected a final list of 30 verb pairs (Tab.12). The selection covered a range of pair types, e.g. some pairs where both verbs are fairly common, some "extreme" pairs where one verb is extremely infrequent compared to its counterpart, among others.

[5]Demographics summary can be found at App.E

[6]We use this model for generating the features as it is expected to have seen the most data (as compared to other models in the set) thus minimizing effects due to OOD issues.

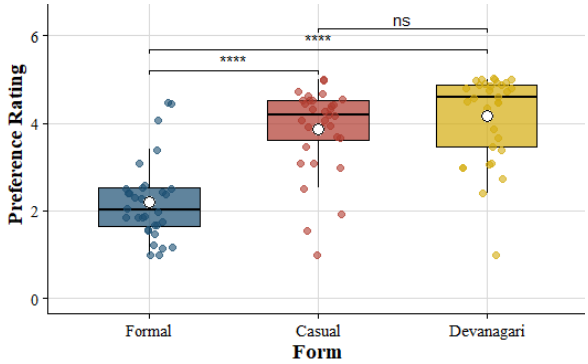


Figure 4: Human Form Preference distribution across 3 orthographic forms. Casual and Devanagari significantly preferred over Formal Latin.

Form: We conducted a preliminary survey to determine which form is preferred by native speakers. 32 participants of varying demographics were shown sentences in all 3 forms, based on a curated set of 5 visually different Hindi verbs [7].

We found that Casual form and Devanagari were almost equally preferred ($p > 0.05$, Fig.4), while Formal Latin form was found unnatural and significantly lowly preferred as compared to the other forms (ANOVA with post-hoc t-tests, $p < 0.05$). This is because most native speakers are unaware of standardization rules which exist for computational consistency.

We finally decide to use Casual form as it is more comparable to the English verbs (both use Latin script) and is natural to native speakers.

4.2.3 Experiment Design

The experiment was implemented using Psytoolkit (Stoet, 2010, 2017). Each participant rated 50 sentence pairs (after 5 practice trials). They rated their preference (7-point scale) between 2 sentences of the same frame and verb pair on the screen, one on the extreme right and one on the extreme left (language sides were counterbalanced, Fig.5)). Verb pairs across frames were counterbalanced among

[7]based on factors such as length, consonant cluster/diacritic/long vowel presence, and if the Formal Latin form contains uppercased phonemes e.g. Dha

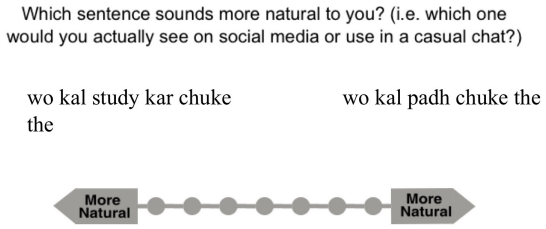


Figure 5: Experiment Screen

participants over 150 sentence pairs. After outlier removal, 699 ratings across 100 sentences, provided by 14 participants, were considered for analysis.

5 Analyses

We present our findings in a series of analyses, beginning with understanding how the data looks like, i.e. human judgment patterns on the verb pairs, and then evaluating model alignment through three connected experiments. Each experiment ends with observations about model–human alignment.

5.1 Human Preferences

From the Human Preference experiments we observed moderate inter-annotator agreement (ordinal Krippendorff's $\alpha = 0.365$, on a 7-point scale), confirming variability in human judgments, in-line with typical ranges in the field as reported by Kodali et al. (2025); Sterner and Teufel (2025). We then used a Monte Carlo simulation test for uniformity to characterize this variation(100,000 iterations; see App.F). Verb pairs with non-uniform distributions ($p < 0.05$) were then manually categorized by distribution shapes (Fig.3)[8]:

1. UN (n=12): Distributions statistically similar to uniform (p > 0.05), indicating no significant preference, e.g. change-बदल
2. HIP (n=13): Strong right-skewed distributions indicating clear preference for Hindi verbs, e.g. tear-फाड़
3. ENP (n=3): Strong left-skewed distributions indicating clear preference for English verbs, e.g. kiss-चूम
4. MP (n=2): Multimodal distributions suggesting divergent speaker groups/context dependence, e.g. imagine-सोच

We see that over half of the verb pairs elicited a strong one-language preference (HIP, ENP) –unsurprisingly, particularly favoring Hindi– while the rest showed fair mixability (UN + MP). This shows that Hinglish do-support verb mixing acceptability lies on a spectrum and is speaker & context dependent in some cases.

These categories also enable us to perform a structured test of model alignment: do LLMs' perplexity ratios distinguish these three patterns without any supervision?

[8]Appendix Tab.14 lists specific verb pairs in each category.

Model Name	Accuracy	Macro-Precision	Macro-Recall	Macro-F1
Gemma3-1b	0.37	0.24	0.28	0.21
Gemma3-4b	0.37	0.22	0.28	0.22
Gemma3-12b	0.37	0.30	0.28	0.22
Llama3.1-8B	**0.40**	**0.36**	**0.39**	**0.33**
Llama3.1-70B	0.33	0.20	0.25	0.20
Sarvam-1	0.20	0.28	0.32	0.20
Qwen3-32B	0.27	0.31	0.37	0.26
Random Baseline	0.27	0.29	0.37	0.27

Table 2: Exp 1 metrics. Bold denotes column-wise best across models.

5.2 Exp1: Do LLMs model Human Variation in Hindi-English Verb Code-Mixing?

This experiment represents the direct hypothetical use case: a linguist using an LLM as a proxy for human judgments of linguistic variation on novel items (code-mixed verb pairs in Hinglish).

Setup: For each verb pair, we conducted a majority vote across all 15 conditions (5 frames × 3 orthographic forms) based on their perplexity ratios (PR_θ)/preferences (Sec.4.1). If a majority of the conditions (more than 8) showed the same language preference we label the verb as HIP/ENP, and Mixable otherwise. This yielded model predictions for all 30 verb pairs, which we compared against human categories using 3-way classification metrics.

Results: Tab.2 shows that:

- **All models perform at or below the random baseline** (Macro-F1 = 0.27) where none are statistically significantly different from baseline[9]. Only Llama3.1-8B marginally exceeds baseline (F1 = 0.33), but this is far from useful on a novel linguistic variation study.
- **There is no positive model size effect**: Scaling within the Gemma does not change performance, while scaling Llama drops performance. Across families, Llama 70B per-

[9]McNemar's test on aggregated predictions after Bonferonni corrections, $p > 0.007$

forms identically to the much smaller Sarvam 2B (F1 = 0.20). Note that this is surprisingly **not consistent with** Sterner and Teufel (2025)'s findings of scaling in Llama making it more aligned to human judgments in *code-switching*.

- **Pretraining language mixes do not help** as Sarvam-1, the only model pretrained from scratch on substantial Indic language mixtures, performs the worst. Paradoxically, Qwen3-32B with a special focus towards Chinese, performs near baseline despite its size and limited Hindi exposure.

Model behavior patterns: Examining individual predictions reveals that models exhibit *strong but arbitrary biases*. (refer B for confusion matrices) Llama and Gemma families systematically prefer Hindi across most verbs, while *Sarvam paradoxically prefers English* opposite to its training composition change . Qwen shows more balanced predictions but with no systematic relationship to human judgments. These patterns indicate models encode consistent but arbitrary biases that *neither reflect training distributions nor align with human acceptability*.

Implication: Unsupervised LLMs cannot model human linguistic variation as observed when comparing perplexity ratios to human Hinglish verb mixing preferences. A linguist asking "which mix sounds more natural?" on a similar phenomenon might receive responses no better than chance.

However, like Kodali et al. (2025), would LLMs supervised on some human judgments reveal latent abilities to model them? We do a more intensive study with Exp2(a) and Exp2(b).

5.3 Exp2(a): Supervised 4-way classification with Optimal Feature selection

This experiment represents the use case of an NLP researcher leveraging LLMs as a proxy for human judgments of linguistic variation on novel items given access to some labeled gold data. If models contain systematic linguistic knowledge about Hinglish verb mixing acceptability, this oracle access to labels should reveal it.

Exp1 showed LLMs cannot model linguistic variation in our phenomenon without help. We now test if we can use human labels to identify informative LLM features, and then train classifiers using features to predict verb mixability preference types.

Config	Accuracy	Macro-Precision	Macro-Recall	Macro-F1
All Selected Features	0.27	0.14	0.20	0.16
Sarvam dropped	0.28	0.19	0.21	0.20
Best for Llama3.1-8B	**0.43**	**0.43**	**0.53**	**0.43**
Best for Llama3.1-70B	0.33	0.34	0.40	0.34
Best for Sarvam-1	**0.43**	0.41	0.35	0.34
Best for Qwen3-32B	0.33	0.28	0.30	0.29
Random Baseline	0.25	0.24	0.29	0.25

Table 3: Exp2(a): Four-way classification metrics across configurations. Bold denotes column-wise best across models.

Feature Setup: We use perplexity ratios obtained from the 15 conditions as 'features'. We then compute Spearman correlations between each of the 15 features and human ratings. Note that we restricted this analysis to sentences from HIP and ENP verb pairs (n=52), where humans prefer one language clearly(Appendix Tab.10). We treat human preference on Casual form for any sentence, as the gold counterpart to all 3 forms we obtained the LLM preference on.

Features Selection: The correlation (based on a grid of 7 (models) x 15 (features)) analysis revealed substantial heterogeneity across models. We prune features relating to Gemma models, Formal and Casual Latin form, from the feature set as they showed predominantly positive correlations [10].

For the remaining models[11], we selected the four frames showing most consistent negative correlations (all except "*X(kar)na aasaan hai*") in the Devanagari script.[12]

Classification: We trained L2-regularized logistic regression classifiers to predict all four human categories from the selected features. We used balanced class weights to address category imbalance, grid search for regularization strength, and standard scaling. Results are computed over 10 repeated random sampled 80:20 train-test splits, ensuring each class appears in every test set.

Results: Tab. 3 shows that:

- **Performance remains weak** where none of the combinations differ from random baseline

[10] opposite the desired negative direction where lower perplexity should indicate higher preference of Hindi verb

[11] Sarvam-1 showed near-zero correlations across features but was retained to test whether combining weakly-correlated features might yield emergent signal.

[12] This is interesting as it was the only form where English verbs were visibly in a different orthography (Latin) than the Hindi verb (Devanagari) and frame (Devanagari).

($p > 0.008$), notably including the Naive combination of all selected features.

- We then explored whether a particular model's features, on their own, might be more predictive ("Best for X"). **No model size effect emerges again** as Llama3.1-8B achieves the highest F1 of 0.43 (marginal improvement over the baseline, but far from reliable) while Llama 70B under-performs in comparison. Across models, smaller models (Llama 8B, Sarvam (2B)) either match or outperform bigger models (Llama 70B, Qwen 32B).

- Interestingly, Indic language based Sarvam has a high accuracy while scoring a low Macro-F1 failing to classify any `MP` instances.

Interpretation:

For the NLP researcher in this case, even with human-labeled feature selection and exhaustive search, 4-way classification barely exceeds baseline. The optimal strategy to get better than baseline scores required not just feature selection but also model selection, thus increasing the search space and reducing practical utility.

However, this task faces challenges from severe class imbalance (e.g. `HIP` (n=13) vs. `MP` (n=2)). We therefore test the same setup on simplified binary task (Exp2(b)): can LLMs at least detect when the Hindi form is strongly preferred?

5.4 Exp2(b): Binary Classification

We collapse our categories into a binary task: `HIP` (n=13, 43%) versus not-`HIP` (`ENP` + `UN` + `MP`, n=17, 57%) and use the same supervised workflow as Exp2. As this represents a substantial simplification, if LLMs struggle to detect this boundary, it would suggest that they can't be used to model variation in our context. Thus the **Setup** is identical to Exp2, but with binary labels, eliminating class imbalance as a confound.

Results: Tab. 4 shows:

- **Improvement across all selected models** over the previous experiments. Using all selected features yields F1 = 0.72, well above the binary random baseline. Exhaustive search over feature combinations reveals **Sarvam-1** achieves **F1 = 0.81** (the only statistically significant configuration $p < 0.008$).

- Llama3.1-8B, collapses to F1 = 0.30, however Llama 70B achieves a higher F1 in this setting. Qwen remains near-baseline. Across models, Sarvam (2B) outperformed others, leading to **inconsistent size effects** in this setting as well.

Config	Accuracy	Precision	Recall	F1
All Selected Features	0.72	0.71	0.73	0.72
Sarvam dropped	0.65	0.63	0.73	0.68
Best for Llama3.1-8B	0.45	0.41	0.23	0.30
Best for Llama3.1-70B	0.63	0.65	0.57	0.61
Best for Sarvam-1	**0.80***	**0.76**	**0.87**	**0.81**
Best for Qwen3-32B	0.63	0.75	0.40	0.52
Random Baseline	0.47	0.46	0.43	0.45

Table 4: Exp2(b): Binary classification metrics. Bold denotes column-wise best across models. * = statistically significant difference from baseline.

Interpretation:

Sarvam emerges as the strongest performer in supervised settings (near-best in Exp2(a), best in Exp2(b)) after failing unsupervised prediction (Exp1). This suggests its Indic-focused pretraining may encode latent code-mixing knowledge that surfaces only with supervision, consistent with claims about Hindi and code-mixed data in its training corpus.

6 Discussion and Conclusion

6.1 Why Even "Success" Represents Fundamental Limitations?

The Sarvam-1 result might appear to validate supervised approaches. However, this apparent success reveals critical limitations for practical application:

Model-specific and non-generalizable. The optimal feature combination for Sarvam-1 yields F1 = 0.81, but the same workflow produces F1 = 0.30 for Llama3.1-8B and F1 = 0.61 for Llama3.1-70B. There is no principled way to know which model will work for a given phenomenon, which features to extract, or what performance to expect limiting generalization to new code-mixing patterns.

Information loss. Collapsing 7-point ratings into binary labels discards the gradient acceptability, contextual effects, and speaker-level variation that linguists need to understand sociolinguistic factors governing code-mixing.

Circular dependency. The required workflow demands the very human labels LLM-based prediction was meant to replace. If researchers already have labeled data sufficient for feature engineering, it remains unclear whether LLM-based prediction offers advantages over simply collecting additional human judgments richer in capturing gradient.

Optimistic upper bounds. Our exhaustive search (all 16 feature combinations, no held-out validation) represents an upper bound that likely inflates results. The wide performance range (0.30 to 0.81) reveals model-specific behavior rather than reliable extraction methods. Even accepting these optimistic estimates, only Sarvam approaches adequacy still not good enough for LLM-based simulations for linguistic studies.

6.2 Answering the Research Question

Our three experiments tested progressively more accommodating scenarios for using LLMs as linguistic informants. Experiment 1 showed that unsupervised perplexity ratios cannot reliably predict human preferences. Experiment 2 demonstrated that even with supervised feature selection and oracle labels, 4-way classification barely exceeds baseline. Experiment 3 reveals that even with substantial task simplification to binary detection, apparent success is brittle, coarse-grained, and requires extensive human labels for feature engineering. **Do LLMs model human linguistic variation?** For Hindi-English code-mixing: not reliably enough for practical use as linguistic informants. Models contain extractable signals, but recovery requires impractical optimization that defeats the purpose of using LLMs for linguistic research.

6.3 What This Reveals About LLMs and Linguistic Variation

The progression from unsupervised failure to supervised partial success shows models encode latent code-mixing knowledge that remains inaccessible through standard probing. Sarvam-1's superior supervised performance despite small size (2B) suggests Indic-focused pretraining helps extract Hindi-English patterns with human guidance, yet even this specialized model fails unsupervised detection. Neither parameter scaling nor targeted training data provides a solution. Llama3.1-70B shows lower F1 than its 8B variant, and 2B Sarvam matches 32B Qwen. More fundamentally, even perfectly balanced Hindi-English training data would not generalize: each language pair would require specialized corpora, and models would still fail unsupervised detection.

This suggests the problem may lie not just in what models are trained on, but in how they are trained and structured. Standard next-token prediction objectives learn surface correlations in code-mixing patterns without capturing the underlying sociolinguistic factors—domain dominance, speaker attitudes, processing convenience—that generate them. Models also exhibit arbitrary systematic biases (Llama/Gemma preferring Hindi, Sarvam preferring English) inconsistent with training distributions or human judgments, further indicating that current approaches do not align with sociolinguistic reality.

6.4 Implications for LLMs as Linguistic Proxies

LLMs align well with human judgments on formal phenomena with uniform speaker behavior (Jumelet et al., 2025), but struggle with human linguistic variation in our experiments. These findings clearly demonstrate that current LLMs cannot be used reliably for linguistic studies even for languages with high to mid-range resources (English is in class 5 and Hindi is in class 4 in Joshi et al. (2020) classification scheme - the top two classes).

Our findings open several promising research directions. At the phenomenon level, future work should examine whether our results generalize to other code-mixing types such as adjective-noun alternations, discourse markers, or pragmatic particles; and to syntactic alternations like word order variation in mixed clauses. Testing across other bilingual communities (Spanish-English, Arabic-French, Cantonese-English) would clarify whether the patterns we observe are specific to Hindi-English or reflect fundamental limitations in how LLMs handle sociolinguistic variation.

More broadly, our results highlight the need for training paradigms and architectures that develop genuine variational competence. Current next-token prediction objectives optimize over aggregate statistics, but sociolinguistic competence requires sensitivity to speaker identity, community norms, and contextual constraints. This limitation extends beyond linguistic variation: modeling how different communities interpret politeness, express disagreement, or conceptualize abstract concepts requires awareness that multiple valid perspectives exist, not just statistical averages (Saha et al., 2025). Developing models that can recognize, represent, and navigate variation rather than averaging over it is crucial not just for linguistics but for any application where LLMs interact with diverse human populations or serve as proxies for human behavior across cultural and social contexts.

7 Limitations

Sample size and scope. Our study focused on 30 verb pairs selected to span the preference space, limiting generalizability across the lexicon and to other lexical categories. We tested a single phenomenon (do-support-verb alternation) in one language pair (Hindi-English). Whether our findings reflect general LLM limitations or language-specific challenges requires testing across other code-mixing types (adjective-noun alternations, adverbial mixing), bilingual communities (Spanish-English, Arabic-French), and variable linguistic phenomena beyond code-mixing.

Human annotation. Our 14 annotators were recruited from a specific demographic (native Hindi speakers fluent in English, ages 25-61, primarily from central and northern India), which may not represent the full diversity of Hinglish speakers.

Model selection. We restricted analysis to base models to avoid instruction-tuning confounds. Instruction-tuned, RLHF-aligned, or specialized architectures designed for sociolinguistic tasks might surface different signals. Additionally, we could not test closed-source models due to API limitations in extracting token-level probabilities required for perplexity-based analysis.

Methodological constraints. Our perplexity-based approach aligns with training objectives but alternative methods—probing classifiers, representational similarity analysis, or causal interventions—might reveal latent structure not captured by surprisal. Our exhaustive feature search in Experiment 3 lacked a held-out validation set, potentially inflating reported performance. Future methodological work could explore whether multi-task learning or meta-learning approaches offer more robust extraction of sociolinguistic signals.

8 Ethics Statement

Human subjects research. All human data collection followed ethical research practices. Participants were recruited through Prolific with informed consent obtained within the survey. Participation was voluntary, and participants could withdraw at any time. Participants were compensated at an hourly rate of £8 (according to Prolific's recommended pay guidelines) and took approximately 10 to 15 minutes each to complete the survey. No personally identifiable information was collected beyond demographic data necessary for participant screening.

Data release and privacy. We will publicly release anonymized preference ratings, perplexity ratios, and experimental materials. All released data contains no personally identifiable information. Demographic summaries are presented in aggregate form only.

Potential biases. Our participant sample (primarily from central and northern India, ages 25-61) may not represent the full diversity of Hinglish speakers, potentially biasing our characterization of code-mixing preferences toward specific age and regional groups. Our findings should not be interpreted as universal patterns across all Hindi-English bilinguals.

Broader impacts. This work demonstrates limitations of using LLMs as proxies for human linguistic behavior. While our negative findings discourage impractical applications, they also highlight the continued necessity of human participation in linguistic research. Our released resources enable replication and extension but should not be used to make claims about individual speakers' linguistic competence or to enforce prescriptive norms about "correct" code-mixing.

Environmental considerations. Model evaluation required approximately 10 GPU hours on 2x NVIDIA RTX 6000 GPUs. We used existing pretrained models to minimize computational costs and environmental impact.

References

Sarvam AI. 2024. Sarvam 1: The first indian language llm. https://www.sarvam.ai/blogs/sarvam-1.

Lisa P. Argyle, Ethan C. Busby, Nancy Fulda, Joshua R. Gubler, Christopher Rytting, and David Wingate. 2023. Out of one, many: Using language models to simulate human samples. *Political Analysis*, 31(3):337–351.

Lucia Badiola, Rodrigo Delgado, Ariane Sande, and Sara Stefanich. 2018. Code-switching attitudes and their effects on acceptability judgment tasks. *Linguistic Approaches to Bilingualism*, 8(1):5–24.

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.

Pushpak Bhattacharyya. 2010. IndoWordNet. In *Proceedings of the Seventh International Conference on*

Language Resources and Evaluation (LREC'10), Valletta, Malta. European Language Resources Association (ELRA).

Anik Dey and Pascale Fung. 2014. A Hindi-English code-switching corpus. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland. European Language Resources Association (ELRA).

Björn Gambäck and Amitava Das. 2016. Comparing the level of code-switching in corpora. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1850–1855.

Yuan Gao, Dokyun Lee, Gordon Burtch, and Sina Fazelpour. 2025. Take caution in using llms as human surrogates: Scylla ex machina. *Preprint*, arXiv:2410.19599.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The Llama 3 Herd of Models. *Preprint*, arXiv:2407.21783.

Rohit Gupta, Pulkit Goyal, and Sapan Diwakar. 2010. Transliteration among indian languages using wx notation. In *KONVENS*, pages 147–150.

Luke Hewitt, Ashwini Ashokkumar, Isaias Ghezae, and Robb Willer. 2024. Predicting results of social science experiments using large language models. *preprint*.

Yutong Hu, Quzhe Huang, Mingxu Tao, Chen Zhang, and Yansong Feng. 2024. Can perplexity reflect large language model's ability in long text understanding? *ArXiv*, abs/2405.06105.

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

Jaap Jumelet, Leonie Weissweiler, and Arianna Bisazza. 2025. MultiBLiMP 1.0: A massively multilingual benchmark of linguistic minimal pairs. ArXiv:2504.02768 (Preprint).

Shivani Kapania, William Agnew, Motahhare Eslami, Hoda Heidari, and Sarah Fox. 2024. 'simulacrum of stories': Examining large language models as qualitative research participants. *Preprint*, arXiv:2409.19430.

Prashant Kodali, Anmol Goel, Likhith Asapu, Vamshi Krishna Bonagiri, Anirudh Govil, Monojit Choudhury, Ponnurangam Kumaraguru, and Manish Shrivastava. 2025. From human judgements to predictive models: Unravelling acceptability in code-mixed sentences. *ACM Transactions on Asian and Low-Resource Language Information Processing*, 24(9):1–31.

Ashok Kumar. 1986. Certain aspects of the form and functions of hindi-english code-switching. *Anthropological Linguistics*, pages 195–205.

Carol Myers-Scotton. 1997. *Duelling languages: Grammatical structure in codeswitching*. Oxford University Press.

Adithya Pratapa, Gayatri Bhat, Monojit Choudhury, Sunayana Sitaram, Sandipan Dandapat, and Kalika Bali. 2018. Language modeling for code-mixing: The role of linguistic theory based synthetic data. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1543–1553.

Prolific. 2025. Prolific. `https://www.prolific.com`. Online participant recruitment platform.

Sougata Saha, Saurabh Kumar Pandey, and Monojit Choudhury. 2025. Meta-cultural competence: Climbing the right hill of cultural awareness. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8025–8042, Albuquerque, New Mexico. Association for Computational Linguistics.

Igor Sterner and Simone Teufel. 2025. Minimal pair-based evaluation of code-switching. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 18575–18598. Association for Computational Linguistics.

Gijsbert Stoet. 2010. PsyToolkit: A software package for programming psychological experiments using Linux. *Behavior Research Methods*, 42(4):1096–1104.

Gijsbert Stoet. 2017. PsyToolkit: A novel web-based method for running online questionnaires and reaction-time experiments. *Teaching of Psychology*, 44(1):24–31.

Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, Louis Rouillard, Thomas Mesnard, Geoffrey Cideron, Jean bastien Grill, Sabela Ramos, Edouard Yvinec, Michelle Casbon, Etienne Pot, Ivo Penchev, and 197 others. 2025. Gemma 3 technical report. *Preprint*, arXiv:2503.19786.

Leonie Weissweiler, Kyle Mahowald, and Adele Goldberg. 2025. Linguistic generalizations are not rules: Impacts on evaluation of lms. *Preprint*, arXiv:2502.13195.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara

Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, and 3 others. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of EMNLP 2020: System Demonstrations*, pages 38–45. Association for Computational Linguistics.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, and 41 others. 2025. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

A Transliteration

A.1 Formal Transliteration

Formal transliteration follows established Devanagari romanization conventions with systematic mapping rules. We follow the transliteration system in Table 5 for the *Formal Latin* form. Our approach uses chain-of-thought reasoning with six processing stages: decomposition, analysis, mapping, rule application, combination, and verification.

We employed verb spelling consistency rate as our primary quality metric—calculated by measuring identical transliterations when the same verb is substituted across five different sentence frames. This serves as an effective non-gold-standard metric for evaluating rule adherence without requiring manual annotation. This methodology achieved 91.19% consistency with a mean of 1.10 unique spellings per verb (SD=0.37, N=1,044 verbs) providing high confidence in the reliability of outputs. Analysis against established romanization rules identified systematic errors detailed in Appendix A.1.

A.1.1 Prompt Design

Formal transliteration required extensive prompt architecture due to the complexity of Devanagari-to-Latin mapping rules. Despite the rule-based nature of this task, significant prompt engineering was required to achieve reliable outputs, given the complexity of Devanagari orthography and the challenges of handling tokenization boundaries in neural language models. Our development process incorporated the following refinements (refer Figure Box 1 for final prompt):

1. Basic Mapping Rules: Initial prompt included character-to-character correspondences
2. Case Sensitivity: Added explicit rules for errors w.r.t. letter case handling
3. Nasalization Rules: Incorporated rules handling consonant clustering and nasalization
4. Chain-of-Thought Reasoning: Implemented six-step decomposition process (decomposition, analysis, mapping, rule application, combination, and verification)
5. Few-Shot Examples: Added exemplars showing complete reasoning chains (Prompt Box 3)
6. Self-Verification: Included re-verification step for error detection and correction

A.1.2 Error Analysis

Validation of formal transliteration outputs against established Devanagari romanization rules revealed three systematic error categories: (1) nasalization handling errors, particularly with 'm' nasal exceptions (e.g., संभल → "sanbhala" instead of "sambhala"), (2) character merging mistakes during combination steps (e.g., अटका → incorrect reasoning "'a' + 'Ta' + 'kaa' = 'aTaka'" instead of correct "'aTakaa'"), and (3) consonant clustering over-triggering with hallucinated halants creating false clusters (e.g., निपट → model incorrectly identifying प् + ट cluster where none exists). Despite these systematic errors, the distribution reveal exceptional consistency (refer table 6), with 91% consistency rate and a mean value close to 1 (1.10) with a low standard deviation (0.37) indicating sufficient constraint on formal transliteration by the provided rules leading to reliable outputs.

B Model Prediction Confusion Matrices

Tables 8–9 show confusion matrices comparing model predictions (columns) against human preference categories (rows) for all 30 verb pairs in Experiment 1. The dominant diagonal entries would indicate perfect alignment; instead, models show systematic biases toward one language regardless of human preferences.

B.0.1 Frame Skeleton Analysis

Following is the frame skeleton analysis for all the 5 frames when transliterated where the verb gets substituted with X. We take the most popular frame spellings as the final frames using this analysis where the dictionary for each frame/format is structured as the actual spellings as the key with the total instance count as where this spellings were found as the value. This reveals high frame-level

inconsistency, requiring limitation to the top 5 variants per frame due to substantial variability. This variance, despite formal transliteration's rule-based nature, results from computational complexity in the 6-step process involving character-level decomposition and token-level mapping operations. The most frequent variant was selected as the canonical frame for each format. The significant frame-level variance contrasts sharply with non-standardized transliteration's structural consistency, demonstrating the challenges of maintaining uniformity in complex multi-step linguistic processing despite rule-based constraints.

```
Format 1 Skeletons:
{'vo X rahe hain': 804, 'vao X rahe
↪  hain': 114, 'vo X rahe han': 57,
↪  'vao X rahe han': 32, 'vao X rahae
↪  hain': 16, ...}

Format 2 Skeletons:
{'Xnaa aasaan hai': 634, 'Xnaa aasaan
↪  haai': 95, 'Xna aasaan hai': 87,
↪  'Xna aasana hai': 82, 'Xnaa aasana
↪  hai': 38, ...}

Format 3 Skeletons:
{'vo kala X chuke the': 782, 'vao kala X
↪  chuke the': 90, 'vao kala X chukae
↪  thae': 64, 'vo kala X chukae thae':
↪  44, 'vao kala X chuke thae': 23,
↪  ...}

Format 4 Skeletons:
{'Xnaa manaa honaa chaahie': 436, 'Xnaa
↪  manaa honaa chaahiye': 222, 'Xnaa
↪  manaa honaa chaahiie': 70, 'Xna mana
↪  hona chaahiye': 57, 'Xnaa manaa
↪  honaa chaahiiye': 25, ...}

Format 5 Skeletons:
{'vo aaraama se Xte rahenge': 485, 'vao
↪  aaraama sae Xte rahenge': 137, 'vao
↪  aaraama se Xte rahenge': 94, 'vao
↪  araamaa sae Xte rahenge': 60, 'vao
↪  araama sae Xte rahenge': 53, ...}
```

B.1 Non-standardized Transliteration

Non-standardized transliteration produces a popular version of the Devanagari counterpart, capturing how Hindi speakers naturally type Hindi words using standard English keyboards in informal digital contexts. This approach generates phonetic Latin representations reflecting authentic user behavior rather than prescribed schemes. We achieved a 47.03% consistency rate with a mean of 1.559 unique spellings per verb (SD=0.76, range=1-5). This moderate consistency reflects the inherent variability expected in non-standardized romanization practices. Frame Skeleton analysis of the most frequent frame patterns selected canonical frames for subsequent experiments, ensuring stimuli reflect authentic usage patterns based on popularity.

B.1.1 Verb Consistency Analysis

Since non-standardized transliteration captures popular usage patterns rather than rule-based accuracy, outputs were accepted based on frequency and consistency patterns. Analysis revealed systematic variations: (1) vowel length inconsistencies (e.g., "aasan" vs "asan"), (2) consonant doubling variations (e.g., "chuke" vs "chukke"), and (3) phonetic approximations of retroflex sounds, reflecting authentic user behavior in digital contexts. The verb consistency distribution (refer Table 11) exhibits moderate variability characteristic of non-standardized systems. While 25% of verbs achieve perfect consistency (single spelling), the median of 2.0 indicates that typical verbs generate two distinct spellings across frames. The higher standard deviation (0.76) compared to formal transliteration reflects authentic user behavior variability, where personal typing preferences and phonetic interpretations influence output. Notably, both approaches share the same maximum variance (5 unique spellings), but this represents different underlying causes: computational errors in formal transliteration versus legitimate alternative representations in non-standardized transliteration. The 47.03% consistency rate, while lower than formal systems, indicates substantial convergence toward popular spellings, validating the existence of informal but recognizable standards in digital Hindi typing practices.

B.1.2 Frame Skeleton Analysis

Similar to Formal transliteration following is the frame skeleton analysis for Non-standardized transliteration demonstrating exceptional structural consistency. The dominant frames achieve near-perfect consistency (>97% across all formats), validating that structural elements are more standardized than lexical elements in non-standardized transliteration.

```
Format 1 Skeletons:
{'wo X rahe hain': 1044}

Format 2 Skeletons:
{'Xna aasan hai': 1036, 'X na aasan hai':
↪  5, 'Xnaa aasan hai': 3}

Format 3 Skeletons:
{'wo kal X chuke the': 1042, 'wo Xal k
↪  chuke the': 1, 'wo kaX l chuke the':
↪  1}
```

```
Format 4 Skeletons:
{'Xna mana hona chahiye': 1037, 'X na
↪  mana hona chahiye': 6, 'Xnā mana
↪  hona chahiye': 1}

Format 5 Skeletons:
{'wo aaram se Xte rahenge': 1022, 'wo
↪  aaram se Xthe rahenge': 5, 'wo aaram
↪  se X te rahenge': 3}
```

C Verb list pre-processing

1. From all entries, we picked the ones POS tagged as the main verb: `V`, `V.VINT`, etc. This resulted in a list of 34,510 entries out of 136,154 total.

2. We cleaned the list further to obtain verb roots by removing:

   - **Phrasal verbs**: entries with spaces on either the Hindi or English sides,
   - **Compound verbs/Inflected verb forms**: entries with "–" or "," (such as गा, गे, गीई),
   - **Transliteration errors**: such as बनाअ.

D Model details

E Participant Demographics

Participants' ages ranged from 25 to 50 (mean 35.9), with one participant aged 61. Most participants were educated in English-medium institutions, with a few having Hindi-, Gujarati-, or Marathi-medium education. All participants were originally from central or northern states of India, and a few had recently lived abroad (e.g., in the UK, Canada, or USA). Nine participants reported Hindi as their only native language, while the remaining five were native bilinguals (Hindi–Marathi or Hindi–English).

F Testing Uniformity of Human Responses

We tested the null hypothesis H_0 that the Likert-scale responses were drawn from a uniform distribution across 7 categories. Let N be the total number of responses and $O = (O_1, \ldots, O_7)$ the observed counts. Under H_0, each category has probability $p_i = 1/7$, and the probability of observing O is given by the multinomial PMF:

$$P(O \mid H_0) = \frac{N!}{O_1! \ldots O_7!} \prod_{i=1}^{7} p_i^{O_i}.$$

Because N was small, we used a Monte Carlo simulation to approximate the exact p-value: we generated $100,000$ datasets of size N from the uniform multinomial distribution, computed the likelihood for each, and estimated the p-value as the proportion of simulated likelihoods less than or equal to the observed likelihood. We used a significance level of $\alpha = 0.05$ to determine statistical significance. This approach avoids large-sample approximations and provides an exact small-sample test of uniformity. Once we segregate uniform (`UN`) distributions we move to manually labelling the other samples as either single left peak (`ENP`, English preferred), single right peak (`HIP`, Hindi preferred) or Multiple (`MP`) peaks.

Hindi	Formal Latin
क	ka
ख	kha
ग	ga
घ	gha
च	cha
छ	chha
ज	ja
झ	jha
ट	Ta
ठ	Tha
ड	Da
ढ	Dha
त	ta
थ	tha
द	da
ध	dha
न	na
प	pa
फ	fa
ब	ba
भ	bha
म	ma
य	ya
र	ra
ल	la
व	va
श	sha
स	sa
ह	ha
ड़	Ra
ढ़	Rha
अ	a
आ	aa
इ	i
ई	ii
उ	u
ऊ	uu
ए	e
ऐ	ai
ओ	o
औ	au

Table 5: Devanagari to Formal Latin transliteration mapping.

Statistic	Value
Count	1044
Mean	1.10
Std	0.37
Min	1.00
25%	1.00
50%	1.00
75%	1.00
Max	5.00
Consistency Rate (%)	91.19

Table 6: Distribution of Unique Spellings over 5 frames per Formal Transliterated verb

Feature ID	Feature Combination	English Frame	Hindi Frame
0	Frame 0, Translit Casual	wo X kar rahe hain	wo X rahe hain
1	Frame 0, Translit Devanagari	वो X कर रहे हैं	वो X रहे हैं
2	Frame 0, Translit Formal	vo X kara rahe hain	vo X rahe hain
3	Frame 1, Translit Casual	X karna aasan hai	Xna aasan hai
4	Frame 1, Translit Devanagari	X करना आसान है	Xना आसान है
5	Frame 1, Translit Formal	X karanaa aasaan hai	Xnaa aasaan hai
6	Frame 2, Translit Casual	wo kal X kar chuke the	wo kal X chuke the
7	Frame 2, Translit Devanagari	वो कल X कर चुके थे	वो कल X चुके थे
8	Frame 2, Translit Formal	wo kala X kara chuke the	wo kala X chuke the
9	Frame 3, Translit Casual	X karna manaa hona chahiye	Xna mana hona chahiye
10	Frame 3, Translit Devanagari	X करना मना होना चाहिए	Xना मना होना चाहिए
11	Frame 3, Translit Formal	X karanaa manaa honaa chaahie	Xnaa manaa honaa chaahie
12	Frame 4, Translit Casual	wo aaram se X karte rahenge	wo aaram se Xte rahenge
13	Frame 4, Translit Devanagari	वो आराम से X करते रहेंगे	वो आराम से Xते रहेंगे
14	Frame 4, Translit Formal	wo aarama se X karate rahenge	wo aarama se Xte rahenge

Table 7: Feature frames and templates used in experiments.

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	0	2	1
HindiPreferred (n=13)	1	9	3
MultiPeak (n=2)	0	2	0
Uniform (n=12)	1	10	1
Total	**2**	**23**	**5**

(a) Llama3.1-70B

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	1	2	0
HindiPreferred (n=13)	1	10	2
MultiPeak (n=2)	0	2	0
Uniform (n=12)	1	10	1
Total	**3**	**24**	**3**

(b) Llama3.1-8B

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	2	0	1
HindiPreferred (n=13)	7	3	3
MultiPeak (n=2)	1	1	0
Uniform (n=12)	5	4	3
Total	**15**	**8**	**7**

(c) Qwen3-32B

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	2	0	1
HindiPreferred (n=13)	8	3	2
MultiPeak (n=2)	1	1	0
Uniform (n=12)	9	2	1
Total	**20**	**6**	**4**

(d) Sarvam-1

Table 8: Confusion matrices for Llama, Qwen, and Sarvam models. Columns show model predictions; rows show human preference categories. Note Llama's strong Hindi bias (23-24 Hindi predictions), Sarvam's paradoxical English bias (20 vs. 6), and Qwen's more balanced but still misaligned predictions.

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	0	3	0
HindiPreferred (n=13)	1	10	2
MultiPeak (n=2)	0	2	0
Uniform (n=12)	0	11	1
Total	**1**	**26**	**3**

(a) Gemma3-1B

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	0	2	1
HindiPreferred (n=13)	1	10	2
MultiPeak (n=2)	0	2	0
Uniform (n=12)	1	10	1
Total	**2**	**24**	**4**

(b) Gemma3-4B

Human Category	Eng	Hin	Mix
EnglishPreferred (n=3)	0	2	1
HindiPreferred (n=13)	3	10	0
MultiPeak (n=2)	0	2	0
Uniform (n=12)	0	11	1
Total	**3**	**25**	**2**

(c) Gemma3-12B

Table 9: Confusion matrices for Gemma model family. All variants show strong Hindi bias (24-26 Hindi predictions out of 30 total), with no scaling effect across model sizes.

Feature ↓	**Qwen3-32B**	Gemma3-12b	Gemma3-1b	Gemma3-4b	**Llama3.1-70B**	**Llama3.1-8B**	**Sarvam-1**
Frame0, Translit Casual	0.079	-0.115	0.103	-0.273	0.430	0.358	-0.188
Frame 0, Translit Devanagari	**-0.564**	-0.345	-0.394	-0.261	**-0.176**	**-0.636**	**-0.394**
Frame 0, Translit Formal	-0.018	-0.103	0.333	-0.164	0.285	-0.006	-0.273
Frame 1, Translit Casual	0.183	0.238	0.250	0.341	0.049	-0.006	0.213
Frame 1, Translit devanagari	-0.445	0.360	0.348	0.463	-0.098	-0.116	-0.067
Frame 1, Translit formal	0.268	0.579	0.470	0.543	0.518	0.183	0.463
Frame 2, Translit casual	0.588	-0.285	-0.418	-0.285	0.285	0.479	0.079
Frame 2, Translit devanagari	**-0.624**	-0.079	-0.345	0.079	**-0.685**	-0.152	0.067
Frame 2, Translit formal	0.515	0.273	0.418	0.285	0.370	0.212	-0.067
Frame 3, Translit casual	0.170	0.000	0.201	0.511	-0.304	-0.304	0.097
Frame 3, Translit devanagari	**-0.340**	0.401	-0.261	0.535	**-0.590**	**-0.310**	0.085
Frame 3, Translit formal	0.085	0.085	0.128	0.188	-0.049	0.079	0.170
Frame 4, Translit casual	0.193	-0.249	-0.354	-0.168	-0.291	0.119	-0.095
Frame 4, Translit devanagari	**-0.459**	-0.119	0.014	0.060	**-0.515**	**-0.515**	**0.340**
Frame 4, Translit formal	0.123	0.053	0.266	0.032	-0.214	0.214	0.119

Table 10: Spearman correlation table between humans and models. Red indicates positive values, green indicates negative values, and bold marks selected cells.

Box 1: Formal Transliteration Prompt | Part A

```
You are a meticulous and rule-based transliteration expert. Your sole task is to transliterate Hindi text
↪  from Devanagari script into the Latin script, following a precise set of rules. You must not deviate
↪  from these rules. Your work is methodical and verified.

# Procedure
Use the following mapping for transliteration.

## Mapping
Devanagari_Latin_mapping

## Rules
- **Rule 1 (Accuracy):** The transliteration must strictly follow the mapping provided. There are no
↪  exceptions. Be true to the source word without adding or omitting any characters or introducing
↪  halants or schwas that are not present in the original word.
- **Rule 2 (Case Sensitivity):** Letter case is critical. Preserve the case as specified in the mapping
↪  table (e.g., 'ट' is 'Ta', 'त' is 'ta').
- **Rule 3 (Consonant Clusters):** When a consonant is followed by a halant (्) and another consonant, this
↪  forms a consonant cluster. In clusters, only the final consonant gets the inherent 'a' sound.

Process:
Consonant + halant (्) → just the consonant sound (no 'a')
Final consonant in cluster → gets the 'a' sound (or whatever vowel follows)

Examples:
"चिल्ला" = चि + ल् + ल + आ = chi + l + la + a = chillaa
"नमस्ते" = न + म + स् + ते = na + ma + s + te = namaste
"अच्छा" = अ + च् + छ + आ = a + ch + chha + a = achchhaa
- **Rule 4 (Nasalization):** All nasals, including the bindu (ं), chandrabindu (ँ), and other nasal forms
↪  (like ङ, ञ, ण, ण्) must be transliterated as 'n'. The only exception to this rule is when the nasal is
↪  an 'm', there you may use 'm'. For example, "हंस" is "hansa", and "ढूँढ" is "DhuunDha" but for "कांप" it
↪  is "kaampa" and not "kaanpa".
- **Rule 5 (Examples):** Observe the examples below to understand the nuances.

### Examples
- Input: अटक
  Output: अ (a) + ट (Ta) + क (ka) = aTa + ka = aTaka
- Input: उकसा
  Output: उ (u) + क (ka) + सा (sa + a) = uka + saa = ukasaa
- Input: उछल
  Output: उ (u) + छ (chha) + ल (la) = uchhala = uchhala
- Input: ओढ़
  Output: ओ (o) + ढ़ (Rha) = o + Rha = oRha
- Input: ढूँढ
  Output: ढ (Dha) + ऊ (uu) + ँ (n) + ढ (Dha) = Dhuun + Dha = DhuunDha
- Input: फांद
  Output: फ (fa) + आ (aa) + ं (n) + द (da) = faan + da = faanda
- Input: लौटा
  Output: ल (la) + औ (au) + ट (Ta) + आ (aa) = lau + Taa = lauTaa
- Input: बांट
  Output: ब (ba) + आ (aa) + ं (n) + ट (Ta) = baan + Ta = baanTa
- Input: खटखटा
  Output: ख (kha) + ट (Ta) + ख (kha) + ट (Ta) + आ (aa) = khaTa + kha + Taa = khaTakha + Taa =
  ↪  khaTakhaTaa

# Steps
1. **Analyze and Decompose:** Read the input Devanagari word along with the transliterated sentence
↪  provided where the word will be substituted for X. Break the word down into its fundamental units as
↪  individual characters or syllables or consonant-vowel unit. Identify any consonant clusters. For
↪  "नमस्ते", the units are "न", "म", "स्", "ते".
2. **Initial Mapping match:** Transliterate each unit using the provided `Mapping` table. For "नमस्ते", "न"
↪  -> "na", "म" -> "ma", "स्" -> "s" and not "sa", "ते" -> "te".
3. **Apply Rules and Combine:** Combine the transliterated units two at a time iteratively and make sure
↪  that the border between the units is also joined as per the rules and examples provided. This step is
↪  highly error-prone; be extremely careful. For this example, na + ma + s + te = nama + ste = namaste.
4. **Verification:** The output after Step 3 is your candidate output. Now, verify it against **all**
↪  rules (1-6) one by one by breaking it down into constituent units. Create a mental checklist. Is the
↪  mapping correct? Is the case correct? Are clusters handled? Are nasals 'n' or 'm'? Are word endings
↪  correct? If you find any error, correct it and re-verify. For this example, splitting "namaste" into
↪  its components, we have "na", "ma", "s", and "te". Verifying the new components with the original
↪  mapping from step 2, we can see all components match the mapping. This transliteration is correct.
5. **Final Output:** After thorough verification, construct your final transliteration output.
6. **Confidence Level:** Determine your confidence in the output. If you are certain that the output is
↪  correct and follows all rules, mark it as "Certain". If you have any doubts or if there are edge cases
↪  that might not be covered by the rules, mark it as "Uncertain".

--continued--
```

Box 2: Formal Transliteration Prompt | Part B

```
## Output Format
The output should be a valid JSON object with the following structure

```json
{
    "step 1": <Follow the step 1 instructions>,
    "step 2": <Follow the step 2 instructions>,
    "step 3": <Follow the step 3 instructions>,
    "output": <The transliterated text in Latin script from step 3 copied identically>,
    "step 4": <Follow the step 4 instructions and verify the output against the rules very thoroughly>,
    "final_output": <The final output after verification>,
    "confidence": <"Certain" or "Uncertain">
}
``
```

Box 3: Few-Shot Example in the Formal Transliteration Prompt

```
***Few-Shot Example for Formal Transliteration**

*User Input:*
Input Hindi (Devanagri) Word: 'छुप'
Transliterated Sentence where word will be substituted: 'Xnaa manaa honaa chaahie'

*Assistant Output:*
```json

    "step 1": "Analyze and Decompose: The input word is 'छुप' which will be substituted into the sentence
    'Xnaa manaa honaa chaahie' post transliteration. Breaking it down: 'छ' + ु' + 'प'. 'छ' is the
    consonant 'chha' with the vowel sign ु' (short 'u'). 'प' is the consonant 'pa'. There are no
    consonant clusters or nasalization.",
    "step 2": "Initial Mapping match: 'छ' + ु' = 'chhu' (from 'छ' = 'chha', but with ु' it becomes
    'chhu'). 'प' = 'pa'.",
    "step 3": "Apply Rules and Combine: Combine the transliterated units: 'chu' + 'pa' = 'chupa'. There
    are no consonant clusters, so no schwa dropping is needed. No nasalization is present.",
    "output": "chhupa",
    "step 4": "Verification: Breaking down the final output `chhupa` into constituents: chhu + pa.
   - Rule 1 (Accuracy): 'chhu' is 'छ' with ु' , 'pa' is 'प'. Both match the mapping and combine into
    'छुप'.
   - Rule 2 (Case Sensitivity): All lowercase as per mapping.
   - Rule 3 (Consonant Clusters): No clusters present.
   - Rule 4 (Nasalization): No nasal present.
   - Rule 5 (Examples): Structure matches similar words in the examples.
    All rules are satisfied.",
    "final_output": "chhupa",
    "confidence": "Certain"

```
```

Box 4: Casual Transliteration Prompt

```
Your task is to transliterate Hindi text written in the Devanagari script into a casual, phonetic Latin
     letters.

The output should be easy to read for an English speaker and should reflect how a native Hindi speaker
     might type Hindi words using a standard English keyboard.

# Rules
1. **No Diacritics:** Do not use any special characters, accents, or diacritics (e.g., use 'a', not ā'').
2. **Phonetic Spelling:** Use common English letters that best represent the Hindi sound.

# Task
Now, transliterate the following Hindi text. Only provide the transliterated text as the output.
```
```

Statistic	Value
Count	1044
Mean	1.559
Std	0.76
Min	1.00
25%	1.00
50%	2.00
75%	2.00
Max	5.00
Consistency Rate (%)	47.03

Table 11: Distribution of Unique Spellings over 5 frames per Casual Transliterated verb

Hindi verb	English verb
देख्	witness
भगा	speed
भेद	pierce
महक	smell
खौला	boil
लुभा	seduce
पढ	read
पढ	study
चल्	move
खटखटा	knock
भाँप	gauge
घबड़ा	panic
सोच	imagine
पिरो	string
गिन	enumerate
छू	touch
बदल	change
ललकार	dare
बजा	ring
फाड़	tear
तरस	yearn
छिड़क	spray
चूम	kiss
रोक	hinder
मिटा	rub
रो	weep
बना	create
ले	collect
फूट	crack
पीस	pulverise

Table 12: Hindi–English verb pairs used in our experiments.

Model	HuggingFace ID	Parameters
Gemma3-1b	`google/gemma-3-1b-pt`	1B
Gemma3-4b	`google/gemma-3-4b-pt`	4B
Gemma3-12b	`google/gemma-3-12b-pt`	12B
Llama3.1-8B	`meta-llama/Llama-3.1-8B`	8B
Llama3.1-70B	`meta-llama/Llama-3.1-70B`	70B
Qwen3-32B	`Qwen/Qwen3-32B`	32B
Sarvam-1	`sarvamai/sarvam-1`	2B

Table 13: Overview of evaluated language models and their characteristics.

Distribution Class	Verb Tuples
SinglePeakRight	(पढ-read), (मिटा-rub), (बना-create), (फाड़-tear), (रो-weep), (पढ-study), (तरस-yearn), (चल्-move), (भगा-speed),(रोक-hinder), (देख्-witness), (गिन-enumerate), (पीस-pulverise)
SinglePeakLeft	(खौला-boil), (चूम-kiss), (महक-smell)
MultiPeak	(बजा-ring), (सोच-imagine)
Uniform/Random	(ले-collect), (छू-touch), (बदल-change), (खटखटा-knock), (छिड़क-spray), (भाँप-gauge), (भेद-pierce), (पिरो-string), (घबड़ा-panic), (लुभा-seduce), (फूट-crack), (ललकार-dare)

Table 14: Verb distribution by class