Generating and Analyzing Disfluency in a Code-Mixed Setting

Aryan Paul
Jadavpur University
Kolkata, India
aryanpauljubcse25
@gmail.com

Tapabrata Mondal
Jadavpur University
Kolkata, India
tapabratamondal
@gmail.com

Dipankar Das
Jadavpur University
Kolkata, India
dipankardipnil2005
@gmail.com

Sivaji Bandyopadhyay
Jadavpur University
Kolkata, India
sivaji.cse.ju
@gmail.com

Abstract

This work explores the intersection of code-mixing and disfluency in bilingual speech and text, with a focus on understanding how large language models (LLMs) handle code-mixed disfluent utterances. One of the primary objectives is to explore LLMs' ability to generate code-mixed disfluent sentences and to address the lack of high-quality code-mixed disfluent corpora, particularly for Indic languages. We aim to compare the performance of LLM-based approaches with traditional disfluency detection methods and to develop novel metrics for quantitatively assessing disfluency phenomena. Additionally, we investigate the relationship between code-mixing and disfluency, exploring how factors such as switching frequency and direction influence the occurrence of disfluencies. By analyzing these intriguing dynamics, we seek to gain a deeper understanding of the mutual influence between code-mixing and disfluency in multilingual speech.

1 Introduction

In this section, we provide essential background on code-mixing and disfluency, setting the stage for our contributions in this study.

1.1 Code-Mixing

Multilingual speakers may unconsciously switch languages due to the interconnected nature of their mental lexicons. Possible reasons include:

- Lack of Language Proficiency (Grosjean, 1982).
- Faster Retrieval (Heredia and Altarriba, 2001).
- Understandability (Heredia and Altarriba, 2001).

Code-Mixing refers to mixing elements from two languages within a single sentence, even down to morpheme level (the smallest units of meaning) (Bhatia and Ritchie, 2008).
Example: *Amake aajke* project presentation *dite hobe, tai* stress *hocche*. (switch within clause)
Translates to: "I have to give a project presentation today, so I'm feeling stressed."

Although certain scholars use the terms code-mixing and code-switching interchangeably, code-switching is often defined differently. Code-switching refers to switching between two languages across sentence or clause boundaries, typically at the word, phrase, clause, or sentence level, but not within a single morpheme (Bhatia and Ritchie, 2008).
Example: I can't come to the party tonight. *Amay kal shokale uthte hobe.* (switch at sentence-boundary)
Translates to: "I can't come to the party tonight. I need to wake up early tomorrow."
Example: She was very late, *tai ami wait korini.* (switch at clause-boundary)
Translates to: "She was very late, that is why I did not wait."

Note: These are Bengali-English code-mixed sentences where Bengali words are italicised.
The dominant language in a code-mixed sentence which provides the grammatical structure (e.g., word order, function words, morphology) is called the matrix language. The less dominant language, which contributes content words (e.g., nouns, verbs) but adapts to the grammar of the matrix language, is called the embedded language. These are defined by Myers-Scotton (1997).

1.2 Disfluency

Disfluencies refer to interruptions or breaks in the smooth flow of spoken language. These are natural parts of speech that occur when a speaker pauses, restarts, or struggles to articulate words, common in spontaneous or unplanned speech. The following are some examples of Bengali-English code-mixed disfluent sentences:

- **Interregnum:** I think, ummm, *pora shuru korte hobe.*
- **Repetition:** *Ami ami jabo* exam *dite.*
- **False Start:** *Ami uh- tui* exam *dili?.*
- **Correction:** Left- sorry Right *e jete hobe.*

The *reparandum* (words supposed to be corrected or ignored), *interregnum* (optional discourse cues), and *repair* are marked and Bengali words are italicised.

1.3 Our Contribution

- We try to mitigate the scarcity of high-quality code-mixed disfluent corpora, particularly for Indic languages, by producing a Bengali-English code-mixed disfluent corpus generated via zero-shot prompting.
- We develop novel metrics to quantitatively assess disfluency phenomena.
- Compare different LLMs (GPT-4o, GPT 4.5-Preview, Gemini 2.0 Flash, Gemini 2.0 Thinking, Gemini 2.5 Pro) with respect to the nature of their generated code-mixed disfluent sentences.
- Investigate novel LLM (GPT-4o) based methods for disfluency detection and compare performance with traditional approaches for the same.
- Explore the interplay between code-mixing and disfluency to understand their mutual influence.

2 Literature Review

Disfluency is a characteristic of spontaneous speech and introduces erroneous words that do not contribute to the semantics of the utterance. Early works surrounding disfluency detection explored a variety of neural architectures. Zayats et al. (2016) utilized bidirectional LSTMs with pattern match features to reduce vocabulary sensitivity during training. Wang et al. (2016) proposed an encoder-decoder attention model to capture long-range dependencies for disfluency detection. Jamshid Lou and Johnson (2017) adopted a Noisy Channel Model enhanced with LSTM language models to score candidate analyses based on fluency. Later, Jamshid Lou et al. (2018) introduced an auto-correlational CNN to capture "rough copy" patterns often found in repair disfluencies. More recent efforts have focused on multilingual and syntactic enhancements. Kundu et al. (2022) explored zero-shot disfluency detection in Indian languages using pretrained multilingual models fine-tuned on English. Ghosh et al. (2022) introduced a span classification model that combines contextual information from transformers and structured syntactic features via graph convolutional networks.

The Switchboard (SWBD) corpus (Godfrey and Holliman, 1993) contains annotated English disfluent sentences from 2,400 telephone conversations among 543 US speakers (302 male, 241 female). Participants did not know each other, and conversations were held on topics from a predetermined list. It is a benchmark for monolingual disfluency studies. Passali et al. (2022) proposed the LARD (Large-scale ARtificial Disfluency), which uses linguistic rules to generate synthetic disfluent text, though its application is limited to monolingual settings. Kundu et al. (2022) defined a rule-based procedure for generating disfluencies from fluent sentences. Kundu et al. (2022) also mention that such synthetic data generated by fixed rules may not fully reflect real-world disfluencies — we find this limitation is also applicable to LARD. Bhat et al. (2023) introduced a large-scale, human-annotated corpus for disfluency correction in Indo-European languages, though it, too, is restricted to monolingual utterances.

In the context of code-mixed data, several datasets exist, albeit not specifically for disfluency detection. Raihan et al. (2023) introduced SentMix-3L for sentiment analysis in Bangla-English-Hindi code-mixed text, while Raihan et al. (2024) released EmoMix-3L for emotion detection in the same language mix. Nayak and Joshi (2022) proposed HingCorpus, a Hindi-English code-mixed dataset in Roman script. Gupta et al. (2024) explored rule-based prompting techniques for large language models (LLMs) to generate code-mixed text. However, none of these datasets were designed with disfluency detection in mind. The ICON (2023) Shared Task dataset provides disfluent code-mixed data specifically for disfluency detection, but upon closer inspection, the semantic quality of the utterances was found to be lacking in resemblance to real-world human speech patterns.

3 Methodology

3.1 Zero-Shot Prompting for Generating Corpus

We propose two distinct outcome-based prompting techniques for generating code-mixed disfluent sentences:

- **Translating Disfluent Sentence to Code-Mixed Setting:** In this approach, we present an English disfluent sentence and request its X-Y code-mixed translation or counterpart, where X and Y are the target languages.

- **Artificial Conversation Generation with Code-Mixing and Disfluency:** This technique involves zero-shot prompting for generating an artificial conversation between N speakers who code-mix between languages X and Y, that should resemble human disfluent speech, with the inclusion of common types of disfluencies such as false starts, repetitions, and corrections.

We build our initial dataset using the first prompting technique:

- Sample 100 random sentences of length < 15, and $> 10\%$ disfluent tokens from the SWBD Corpus (Godfrey and Holliman, 1993). This was done to restrict sentence length and remove sentences with negligible disfluencies.

- Sentence is injected into the prompt to GPT-4o, and result (Code-Mixed Counterpart) extracted using Open-AI's structured output parser.

- Manually label sentences using binary labels, 0 for fluent and 1 for disfluent tokens.

The prompt uses the Self-Consistency (Wang et al., 2023) technique to achieve it's goal. The detailed prompt is excluded to maintain conciseness.
The second prompting technique is used to build the final dataset, that we name, MixFluent.

- The two family of LLMs used are GPT-4 (OpenAI, 2024) (GPT-4o and GPT-4.5 Preview) and Gemini (Gemini-Team, 2024) (Gemini 2.0 Flash, 2.0 Flash Thinking and 2.5 Pro).

- A topic X is chosen randomly from a list of pre-specified topics (Life, Travel, Academics, Sports, Food, Car, Movies, Books) is chosen with decreasing probability in order of mention.

- One of the conditions: telephone or face-to-face conversation is chosen randomly.

- One of the Large Language Models is chosen at random.

- We use OpenAI Platform, and Gemini Platform to prompt the LLM as required.

- The result is parsed into individual sentences by splitting at ".", "?", and "!".

- Each such conversation is chosen at random for manual review and editing. All the conversations are labelled both at token and span level manually.

This is repeated till approximately 2000 sentences are obtained. The detailed prompt is excluded to maintain conciseness.
The MixFluent dataset is manually annotated with a two-tier disfluency labeling scheme. At the token level, we use a scheme similar to one proposed by Shriberg (1994) using the following tags:

- `R1` (reparandum): Words intended to be corrected or ignored.

- `I` (interregnum): Optional discourse cues.

- `R2` (repair): The correction or words supposed to be retained.

In addition, each disfluent span is further categorized into one of the following disfluency types defined as:

- `IR` (Interregnum): A filler word or hesitation marker used to buy time while thinking.
  Example: *Focus korar cheshta korchi* but uh it's impossible.
  Translates to: I'm trying to focus but uh it's impossible.

- RE (Repetition): Repeating a word or phrase unintentionally, often due to planning difficulties or to re-iterate a point or objective.
  Example: *Achha, tahole* maybe um, maybe *amra ek saathe bosh e dekha korte pari*, right
  Translates to: Okay, then maybe, um, maybe we can sit together and meet, right?

- FS (False Start): A prematurely started utterance that is abandoned and replaced.
  Example: *Ami toh ami* actually *bhabchhilam boi kinbo bole.*
  Translates to: I was- I actually was thinking of buying a book.

- CR (Correction): Self-repair of an earlier spoken segment.
  Example: Hmm, *motamoti bujhechi*, but weak entity, sorry, weak relationship *taay doubt ache.*
  Translates to: Hmm, I more or less understood, but weak entity- sorry, I have a doubt in weak relationship.

- PP (Pet Phrase): Frequently used filler expressions that do not add semantic content, often specific to personal habits.
  Example: *Mane, aaj raat e amra ekta* quick revision call *kore ni.*
  Translates to: I mean, let's do a quick revision call tonight.

- ST (Stutter): Repetition of sounds or syllables due to difficulty in speech production.
  Example: *Tui kono* no- notes *baniyechis naki.*
  Translates to: Did you make any no- notes?

- FL (Fluent): A normally produced word without any disfluency.
  Example: *O haan, tor kachhe* indexing *ar* transaction *er* notes *ache.*
  Translates to: Oh yes, do you have the indexing and transaction notes?

All above examples are sourced from the MixFluent Dataset. The *reparandum* (words supposed to be corrected or ignored), *interregnum* (optional discourse cues), and *repair* are marked and Bengali words are italicised.

3.2 Novel Disfluency Metrics

We formulate the two following metrics inspired by language entropy (Gullifer and Titone, 2019):
Fluency Entropy (FE): It is an information-theoretic measure that quantifies the distributional uncertainty of fluency labels across all tokens in a sequence. It evaluates how balanced or skewed the fluency labels are, without considering the order or grouping of tokens.

$$FE = -\sum_{i=1}^{k} p_i \cdot \log_2(p_i)$$

where:

- k is the number of unique fluency labels (e.g., *Fluent, Disfluent*),
- $p_i = \frac{\text{number of tokens with label } i}{\text{total number of tokens}}$

Fluency Entropy (FE) quantifies how evenly fluent and disfluent tokens are distributed across a sequence. A high FE value (close to 1) indicates a balanced presence of fluent and disfluent tokens, suggesting more frequent disfluencies throughout the sentence. A low FE value (close to 0) suggests that one label (typically "fluent") dominates the sequence, indicating that disfluencies are either rare or absent.
Fluency Span Entropy (FSE): It is an information-theoretic metric that measures the diversity of fluency span types in a sequence. Each span is defined as a contiguous segment of tokens with the same fluency label and its length. The entropy is computed over the set of unique *(label, span length)* pairs.

$$FSE = -\sum_{i=1}^{n} p_i \cdot \log_2(p_i)$$

where:

- n is the number of unique $(label, spanlength)$ combinations,
- $p_i = \frac{\text{count of span type } i}{\text{total number of spans}}$

Fluency Span Entropy (FSE) captures the diversity and variability of fluency spans—that is, contiguous segments of tokens sharing the same fluency label and length. A high FSE implies that the sequence contains many different types of fluency spans, both in terms of label and length (e.g., short disfluent interruptions, long fluent stretches, and vice versa). A low FSE, in contrast, suggests uniformity in the fluency structure, with fewer types of spans—either the speech is consistently fluent or disfluencies appear in repetitive, predictable patterns. Together, FE and FSE allow us to assess not only how much disfluency is present (via label distribution), but also how that disfluency is organized and patterned within the sentence.

3.3 Generation Characteristics of LLMs

Each sentence in MixFluent Dataset (generated by an LLM) is represented as 2-D feature using one of:

- Code-Mixing Index and Disfluency Percentage
- Code-Mixing Index and Fluency Entropy
- Language Entropy and Disfluency Percentage
- Language Entropy and Fluency Entropy

where, Language Entropy is defined by Gullifer and Titone (2019), Code-Mixing Index is defined by Gambäck (2014), Fluency Entropy is defined by us in Section 3.2, and Disfluency Percentage is the ratio of disfluent tokens to the total number of tokens in an utterance.

Each sentence generated by the same model is assigned to the same cluster. Cluster evaluation metrics like Silhoutte Score (Rousseeuw, 1987), Davies-Bouldin Index (Davies and Bouldin, 1979) and Calinski-Harabasz Index (Caliński and JA, 1974) are computed to analyse quality of clustering. Additionally, we look into the mean of various metrics computed on the data generated by each model/family to draw further conclusions.

3.4 Few-Shot Prompting for Disfluency Labelling

We propose a few-shot prompting approach for disfluency detection, leveraging the chain-of-thought (CoT) (Wei et al., 2023) and self-consistency (Wang et al., 2023) prompting strategies with GPT-4o (OpenAI, 2024) as the LLM. The detailed prompt is excluded to maintain conciseness. Evaluation is done on the ICON 2023 Shared Task (ICON, 2023) Bengali Test Set. The class-wise F1-Scores are compared against strong supervised baselines like BERT (Devlin et al., 2018), MuRIL (Khanuja et al., 2021) and IndicBERT (Kakwani et al., 2020) fine-tuned using Weighted Cross-Entropy Loss to handle severe class imbalance, Adam with weighted decay optimizer for 20 Epochs each.

3.5 Hypothesis Testing on the Influence of Code-Mixing on Disfluency Patterns

We propose the following hypotheses and test them on the MixFluent Dataset:

- H1: Proximity to a switch triggers disfluency.
- H2: Direction of the switch affects disfluency.
- H3: Increase in switching frequency triggers a higher degree of disfluency.
- H4: The speaker is more likely to be disfluent in the embedded language.
- H5: Disfluent tokens are more likely to be clustered than isolated.

While H1–H4 directly explore the influence of code-mixing on disfluency, H5 investigates the structural distribution of disfluent tokens.

Note: In a code-mixing environment, the switch of codes within a single sentence, even down to morpheme level, has been referred to as a ***switch****. The token which differs in language from the previous token is considered to be a* ***switch point****.*

To test each hypothesis, count disfluent vs fluent tokens

- with & without code-switch in ±2 token window.
- for each direction of code-swithcing in previous token.
- with 0, 1, 2+ switches in ±2 token window.
- for each language label.

We also count disfluent tokens that are isolated or clustered (±1 window). We build contingency tables, compute % disfluency per category, and test independence using χ^2 test.

4 Results and Discussion

4.1 Zero-Shot Prompting for Generating Corpus

We make the following qualitative observations while creating the **initial dataset**, by translating disfluent English sentences sourced from SWBD (Godfrey and Holliman, 1993) corpus, to their Bengali-English counterpart:

- Semantic coherence of non-disfluent spans in the SWBD corpus is relatively poor, which affects the quality of code-mixed outputs as the LLM outputs inherit these deficiencies from the input data.
- The type of disfluency present in the input English sentence preserved in the generated Bengali-English code-mixed sentence.

- The position of disfluent spans often shift in the generated output to cater to the natural syntactic and grammatical constraints of Bengali.
- The positional shift prevents direct transfer of disfluency labels from the input to the output, necessitating manual annotation.

Model	RE	FS	CR	ST	PP	IR
GPT-4o/mini	142	8	12	11	167	480
GPT-4.5 Preview	430	27	40	0	215	577
Gemini 2.0 Flash	202	11	0	6	31	115
Gemini 2.0 Flash Thinking	134	3	3	0	0	104
Gemini 2.5 Pro	305	17	11	7	24	330

Table 1: Class-wise model-wise frequency of generated disfluencies normalized to 1k sentences per model.

According to Table 1, we make the following observations concerning the **MixFluent dataset**:

- Only GPT-4o/mini and Gemini 2.5 Pro successfully followed the prompt to include disfluencies from all classes.
- Gemini 2.0 Flash Thinking and GPT-4.5 Preview produced the least and most disfluencies per 1000 sentences respectively.
- Interregnum (IR) and Stutter (ST) were the classes with highest and least frequency.

The following qualitative observations are made while reviewing and labelling the MixFluent Dataset:

- Occasionally, a third language term “yaar” (from Hindi) introduced into the conversation, even though there was no mention of Hindi mixing.
- The LLMs assume that Bengali is the base language and English is the embedded language.
- Although LLMs generated unnecessary disfluenices at few occasions, the nature of produced text represent real-world disfluencies better than data generated by previous rule-based methods proposed by Passali et al. (2022) and Kundu et al. (2022).

The MixFluent dataset statistics and detailed descriptive metrics are provided in Table 2 and Table 3 respectively.

Statistic	Value
Total Sentences	2049
LLMs Used	5
Average Sentence Length (tokens)	6.48 ± 5.00
Total Tokens	13285
Disfluent Tokens	2990
% Disfluent Tokens	22.51
Mean % Disfluent Tokens per Sentence	18.58%

Table 2: Dataset statistics for the MixFluent (Bengali-English code-mixed disfluent) speech corpus.

Feature	Mean	Std. Dev.	Median	IQR
CMI	48.37	32.74	50.00	25.00–75.00
Language Entropy	0.60	0.42	0.81	0.00–0.95
Fluency Entropy	0.34	0.42	0.00	0.00–0.81
Fluency Span Entropy	0.59	0.74	0.00	0.00–1.00
Disfluency %age	18.58	26.59	0.00	0.00–33.33

Table 3: Descriptive statistics of key features in the MixFluent Dataset. (IQR refers to Inter-Quartile Range)

F1	F2	SIL	DBI	CHI
CMI	Disfl. %age	-0.069	8.032	47.016
CMI	FE	-0.135	55.683	59.665
LE	Disfl. %age	-0.178	12.647	28.867
LE	FE	-0.138	8.240	33.768

Table 4: Clustering metrics for various feature combinations, where F1 and F2 are Features 1 and 2 respectively. CMI: Code-Mixing Index, FE: Fluency Entropy, LE: Language Entropy, SIL: Silhouette Score, DBI: Davies-Bouldin Index, CHI: Calinski-Harabasz Index.

4.2 Generation Characteristics of LLMs

According to Table 4, negative Silhouette Scores indicate poor clustering approaching random assignment. This implies that no model seems to be biased towards certain patterns of disfluency and degree of code-mixing at the same time.

According to Table 5, GPT-4.5 Preview generated

Model	CMI	LE	Disfl. %	FE	FSE
GPT-4o	38.46	0.68	17.95	0.39	0.72
GPT-4.5 Preview	39.94	0.76	28.97	0.54	1.07
Gemini 2.0 Flash	40.20	0.55	13.10	0.24	0.41
Gemini 2.0 Flash Thinking	63.88	0.47	10.52	0.15	0.25
Gemini 2.5 Pro	57.42	0.56	25.63	0.43	0.65

Table 5: Comparison of Models on Mean CMI, Language Entropy (LE), Disfluency Percentage, Fluency Entropy (FE), and Fluency Span Entropy (FSE) of generated data. Above values are mean of each metric.

a significantly higher %age of disfluency than

GPT-4o. Similarly, Gemini 2.5 Pro generated a significantly higher %age of disfluency compared to other Gemini models.

Model Family	CMI	LE	Mean FE	FSE	Disfl. %
GPT-4	38.74	0.70	0.42	0.79	20.03
Gemini	54.42	0.53	0.29	0.46	17.67

Table 6: Mean of metrics calculated on data generated by GPT and Gemini model families.

According to Table 6 GPT-4 models generated higher disfluency percentage, while Gemini models generated a much higher average code-mixing index.

4.3 Few-Shot Prompting for Disfluency Labelling

	F1 Score			
Class	**BERT**	**MuRIL**	**IndicBERT**	**GPT-4o**
Alteration_R	0.46	0.69	0.72	0.00
O	0.87	0.95	0.98	0.93
edit_R	0.78	0.73	0.60	0.00
false_R	0.15	0.30	0.26	0.00
filler_R	0.79	0.96	0.92	0.08
pet_R	0.65	0.79	0.86	0.09
repair_R	0.26	0.45	0.66	0.00
repeat_R	0.11	0.66	0.63	0.07
Weighted F1 Score	0.73	0.91	0.95	0.83
Macro Avg F1 Score	0.51	0.69	0.70	0.14

Table 7: F1 scores for various models across disfluency classes (ICON 2023 Shared Task Bengali Test Set)

While comparing traditional methods to few-shot prompting based disfluency labelling, the following observations (Refer Table 7) are made:

- The LLM (GPT-4o) struggled to produce same number of labels as input tokens. This highlights its weakness in counting scenarios. The results are padded with 'O' label for computing metrics.
- Average F1-Score of the LLM (GPT-4o) is significantly worse than other models.
- IndicBERT (Kakwani et al., 2020) despite having lesser parameters performs better than MuRIL (Khanuja et al., 2021).
- All the models relatively struggled in labelling the class "false_R" which represents false starts.

4.4 Hypothesis Testing on the Influence of Code-Mixing on Disfluency Patterns

H1: Proximity to a switch triggers disfluency.

	Switch Nearby	No Switch Nearby
Disfluent	2144	846
Fluent	7060	3232
%age Disfluent	23.29	20.74

Table 8: Contingency table showing the relationship between disfluency and proximity to a code-switch (considering neighborhood of 5 tokens).

From Table 8, we see that the probability of disfluency is higher when there is a switch point in a ±2 token window. The χ^2 value is 10.378 which is greater than the critical value of 3.814 (at p=0.05). Thus, we can reject the null hypothesis.
Proximity to a code-switch does trigger disfluency.

H2: Direction of the switch affects disfluency.

	Switch into ENG	Switch into BEN
Disfluent	556	515
Fluent	1737	1775
%age Disfluent	24.24	22.48

Table 9: Contingency table showing the relationship between disfluency and switching direction in previous token.

From Table 9, we see that there is a difference of 1.76% in disfluency occurrence for the two possible directions of switch in previous token. The χ^2 value is 1.882 which is much lesser than the critical value of 3.814 (at p=0.05). Thus, we can not reject the null hypothesis.
Different code-switching direction does not trigger disfluency to different extent.

H3: Increase in switching frequency triggers a higher degree of disfluency.

	0 Switches	1 Switch	2+ Switches
Disfluent	846	1005	1139
Fluent	3232	3557	3503
%age Disfluent	20.74	22.03	24.54

Table 10: Contingency table showing the relationship between disfluency and switching frequency in a neighborhood of 5 tokens.

From Table 10, we see that the %age of disfluency

increases with increase in number of switches. The χ^2 value is 18.814 which is much higher than the critical value of 5.9915 (at p=0.05). Thus, we can reject the null hypothesis.
Increase in code-switching frequency triggers a higher degree of code-switching.

H4: The speaker is more likely to be disfluent in the embedded language.

	ENG	**BEN**
Disfluent	1828	1162
Fluent	4852	5440
%age Disfluent	27.95	17.60

Table 11: Contingency table showing the relationship between disfluency and language.

Table 11 shows 10% more disfluency occurence across English words than Bengali. The χ^2 value is 180.929, which is much higher than the critical value of 3.814 (at p=0.05). Thus, we can reject the null hypothesis.
Disfluency is more likely to occur in the embedded language, which is also the speaker's second language in most cases.

H5: Disfluent tokens are more likely to be clustered than isolated.

	Count
Clustered	2552
Isolated	438
%age Clustered	85.32

Table 12: Table showing the frequency of clustered and isolated disfluent tokens.

From Table 12, we see that 85.32% of disfluent tokens appear close to other disfluent tokens (within one word before or after), while only 14.68% occur on their own. This clear difference suggests that disfluencies usually do not happen randomly. Instead, once a speaker becomes disfluent, they are more likely to continue making mistakes for a short time. This could happen because the speaker is thinking hard, struggling to find the right words, or feeling unsure while speaking. This finding is in line with the work of Shriberg (1994), who introduced the concept of ”disfluency clusters” or ”disfluency islands.”
Disfluent tokens are more likely to occur in clusters than isolated.

5 Conclusion

This research provides key insights into the interplay between disfluency and code-mixing in multilingual settings. The major conclusions drawn from this study are as follows:

- Prompt-based generation proves to be a viable strategy for creating disfluent code-mixed conversational data, although it currently requires manual annotation for accuracy.
- Large Language Models (LLMs) underperform in token classification tasks compared to traditional BERT-like architectures.
- Novel metrics, such as Fluency Span Entropy (FSE), offer deeper insights into the structural distribution of disfluent tokens within utterances, beyond simple frequency-based analysis.
- The GPT-4 family of models generated a higher average %age of disfluencies than Gemini models, while the latter generated significantly higher average code-mixing index.
- Experimental evidence reaffirms that code-mixing significantly influences the emergence of disfluencies and may act as a trigger in conversational contexts.

6 Future Work

Building on the findings and limitations of this work, several directions can be explored in future research:

- Extend the **MixFluent** dataset to include other Indic language pairs to facilitate broader cross-linguistic analysis of code-mixed disfluencies.
- Explore how **Large Language Models (LLMs)** can be better utilized for disfluency correction and detection, especially in multilingual and low-resource settings.
- Design and evaluate more **innovative metrics** beyond entropy-based measures to capture the complexity and structure of disfluency and code-mixing.
- Test current and possibly additional **hypotheses** on an expanded dataset to assess their validity across different language pairs and sociolinguistic contexts.

Acknowledgement

This work was supported by Defence Research and Development Organisation (DRDO), New Delhi, under the project "Claim Detection and Verification using Deep NLP: an Indian perspective".

References

Vineet Bhat, Preethi Jyothi, and Pushpak Bhattacharyya. 2023. DISCO: A large scale human annotated corpus for disfluency correction in Indo-European languages. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 12833–12857, Singapore. Association for Computational Linguistics.

Tej Bhatia and William Ritchie. 2008. *The Handbook of Bilingualism*, pages 512 – 546.

Tadeusz Caliński and Harabasz JA. 1974. A dendrite method for cluster analysis. *Communications in Statistics - Theory and Methods*, 3:1–27.

David L. Davies and Donald W. Bouldin. 1979. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-1(2):224–227.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Björn Gambäck. 2014. On measuring the complexity of code-mixing.

Gemini-Team. 2024. Gemini: A family of highly capable multimodal models.

Sreyan Ghosh, Sonal Kumar, Yaman Kumar Singla, Rajiv Ratn Shah, and S. Umesh. 2022. Span classification with structured information for disfluency detection in spoken utterances.

John J. Godfrey and Edward Holliman. 1993. Switchboard-1 release 2. Web Download. LDC97S62.

François Grosjean. 1982. *Life with Two Languages: An Introduction to Bilingualism*. Harvard University Press, Cambridge, MA. Cited by 7033, according to Google Scholar as of 2025.

Jason Gullifer and Debra Titone. 2019. Characterizing the social diversity of bilingualism using language entropy.

Ayushman Gupta, Akhil Bhogal, and Kripabandhu Ghosh. 2024. Code-mixer ya nahi: Novel approaches to measuring multilingual llms' code-mixing capabilities.

Roberto R. Heredia and Jeanette Altarriba. 2001. Bilingual language mixing: Why do bilinguals code-switch? *Current Directions in Psychological Science*, 10(5):164–168.

ICON. 2023. Shared task on disfluency identification, icon, iiit hyderabad.

Paria Jamshid Lou, Peter Anderson, and Mark Johnson. 2018. Disfluency detection using auto-correlational neural networks. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4610–4619, Brussels, Belgium. Association for Computational Linguistics.

Paria Jamshid Lou and Mark Johnson. 2017. Disfluency detection using a noisy channel model and a deep neural language model. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 547–553, Vancouver, Canada. Association for Computational Linguistics.

Divyanshu Kakwani, Anoop Kunchukuttan, Satish Golla, Gokul N.C., Avik Bhattacharyya, Mitesh M. Khapra, and Pratyush Kumar. 2020. IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In *Findings of EMNLP*.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages.

Rohit Kundu, Preethi Jyothi, and Pushpak Bhattacharyya. 2022. Zero-shot disfluency detection for Indian languages. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 4442–4454, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

C. Myers-Scotton. 1997. *Duelling Languages: Grammatical Structure in Codeswitching*. Clarendon Press.

Ravindra Nayak and Raviraj Joshi. 2022. L3cube-hingcorpus and hingbert: A code mixed hindi-english dataset and bert language models.

OpenAI. 2024. Gpt-4o system card.

Tatiana Passali, Thanassis Mavropoulos, Grigorios Tsoumakas, Georgios Meditskos, and Stefanos Vrochidis. 2022. LARD: Large-scale artificial disfluency generation. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 2327–2336, Marseille, France. European Language Resources Association.

Md Nishat Raihan, Dhiman Goswami, Antara Mahmud, Antonios Anastasopoulos, and Marcos Zampieri. 2023. Sentmix-3l: A bangla-english-hindi code-mixed dataset for sentiment analysis.

Nishat Raihan, Dhiman Goswami, Antara Mahmud, Antonios Anastasopoulos, and Marcos Zampieri. 2024. Emomix-3l: A code-mixed dataset for bangla-english-hindi emotion detection.

Peter J. Rousseeuw. 1987. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65.

Elizabeth Shriberg. 1994. Preliminaries to a theory of speech disfluencies.

Shaolei Wang, Wanxiang Che, and Ting Liu. 2016. A neural attention model for disfluency detection. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 278–287, Osaka, Japan. The COLING 2016 Organizing Committee.

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-thought prompting elicits reasoning in large language models.

Vicky Zayats, Mari Ostendorf, and Hannaneh Hajishirzi. 2016. Disfluency detection using a bidirectional LSTM. *CoRR*, abs/1604.03209.