An improved Code-Switching Detection System for some Indic Languages

Karan Bhanushali
University of Stuttgart
Stuttgart, Germany
karanbhanushali314@gmail.com

Fritz Hohl
Sony Europe Limited
Stuttgart, Germany
Fritz.Hohl@sony.com

Abstract

Code-switching is a common feature of multilingual communication, and reliably identifying where the language switches is essential for downstream tasks such as generating code-switched machine translations. This paper introduces CSDI, a Code-Switching Detection (CSD) system for Indic text, which jointly learns CSD, Named Entity Recognition, and Part-of-Speech tagging through a shared encoder. Leveraging multitask learning, CSDI captures linguistic cues that signal switching boundaries and achieves a new state-of-the-art macro-F1 score with near-zero ΔCMI across six Indic languages. The model also demonstrates strong cross-lingual transfer, effectively leveraging high-resource languages to improve low-resource performance. Despite challenges such as intra-word code-mixing and limited token-level context, CSDI establishes a new baseline for scalable, low-resource NLP research in code-mixed environments.

1 Introduction

Code-switching (CS) is the alteration between two or more languages within a single conversation or utterance. This is a natural characteristic of multilingual societies such as India, where speakers frequently mix English with native languages in both spoken and written communications. This phenomenon is especially common in social media, everyday conversations, and in film dialogues.

When automatically translating Indic movie dialogues, this phenomenon cannot be ignored. Analysis of our internal data shows that roughly 15% of the words in Indic movie subtitles are English elements. These mixed-language segments behave differently from pure monolingual ones during translation, making CSD an essential step in translation.

Existing systems for some Indic languages (Bengali, Hindi, Kannada, Malayalam, Marathi, and Telugu) offered not enough accuracy for our requirements, which motivated the development of an improved system that achieves better results on standard test sets and on an internal movie dialogue dataset.

In this work, we only consider code-switching where entire words are considered monolingual; intra-word code-switching, while present, is rarer and requires a different approach from the one proposed here. Also, we assume there is maximally one code-switched language (concretely English) in the source language text. Ignoring more than one code-switched language and languages other than English does not completely reflect the reality in the datasets, but it comes relatively close statistically.

To address the limitations of existing systems, we introduce a multitask learning framework that jointly performs CSD, NER, and POS tagging through a shared encoder. This enables the model to capture richer linguistic cues around switching boundaries, leading to substantial improvements over prior work. The resulting system achieves new state-of-the-art performance across six Indic languages, exhibits near-zero ΔCMI, and transfers effectively from high-resource to low-resource settings.

2 Background

Code-switching has evolved from being viewed as a linguistic imperfection (Bolonyai, 2009) to a natural feature of multilingual discourse (Grosjean, 2010). Early studies classified CS into inter-sentential, intra-sentential, intra-word, and tag-switching (Poplack, 1981), governed by grammatical and social factors (Muysken, 1995). In India, with 121 major languages and a high bilingualism, CS is widely observed in social media and films, often used for emphasis or identity expression (Dey and Fung, 2014; Pratapa and Choudhury, 2017).

To explain how CS is structured, the Matrix Language Frame (MLF) model (Jake and Myers-Scotton, 1997) distinguishes the dominant (matrix) language from embedded ones. In Indic CS, Hindi–English switches frequently occur at lexical boundaries, typically English nouns and verbs (Dey and Fung, 2014). These linguistic insights highlight where switching boundaries are likely to appear, making them directly relevant to token-level detection.

Beyond understanding the structure of CS, it is also important to quantify its extent. The Code-Mixing Index (CMI) (Das and Gambäck, 2014) is a widely used metric that measures the proportion of tokens belonging to the embedded language within an utterance. Formally, for a sentence with N tokens and n_E tokens from the embedded language, CMI is defined as:

$$\text{CMI} = \frac{n_E}{N} \times 100$$

A CMI value of zero corresponds to purely monolingual text, while higher values indicate increased code-mixing. In this work, we use CMI to characterize dataset mixing intensity and to analyze model behavior under varying degrees of code-switching.

With the growth of NLP for multilingual communities, computational approaches progressed from early rule-based systems to modern Deep Learning methods. CS datasets now exist for several tasks, including POS tagging (Jamatia et al., 2018), Sentiment Analysis (Patwa et al., 2020), and Machine Translation (MT) (Dhar et al., 2018). Token-level Language Identification (LID), a crucial preprocessing step for many CS tasks, has benefited from multilingual encoders such as IndicBERT (Kakwani et al., 2020) and MuRIL (Khanuja et al., 2021), showing that fine-tuning on CS corpora improves performance (Santy et al., 2021). Multitask Learning (MTL) with POS has also been effective in uncovering structural clues (Winata et al., 2018; Pitale and Malapati, 2023).

Recent CSD approaches such as AnE (Sterner, 2024) aggregate CS corpora for multilingual token-level LID and identifies English even in unseen language pairs (e.g., Indonesian–English), achieving 2.3–4.6% F1 gains over pair-specific SoTA using XLM-RoBERTa. While these multilingual approaches demonstrate strong generalization, Indic-specific challenges such as data scarcity and romanization variations remain underexplored.

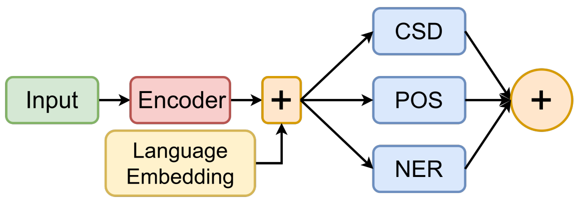


Figure 1: Architecture of CSDI. A shared encoder processes romanized input tokens. A trainable matrix-language embedding is added to the encoder representations to condition them on the base Indic language. The conditioned representations are passed to three task-specific heads (CSD, POS, NER). During training, losses from the three tasks are combined additively and jointly backpropagated to the shared encoder.

Although CS is a relatively frequent phenomenon, obtaining labelled CS data for supervised training is particularly challenging, as naturally mixed utterances are sparsely distributed across large monolingual corpora. Additionally, inconsistent orthography and informal romanization introduce noise and ambiguity, especially in user-generated content. These issues are not fully addressed by existing multilingual LID systems, which often struggle with Indic-specific variation. This highlights the need for more robust and adaptable approaches to token-level detection in romanized Indic–English settings.

3 CSDI – Code-Switching Detection for Indic Languages

In this section, we present CSDI (Code-Switching Detection for Indic Languages), our proposed multi-task learning (MTL) framework designed to detect code-switching in Indic text.

CSDI jointly learns CSD, NER, and POS tagging through a shared encoder. This design allows the model to leverage syntactic and semantic cues from auxiliary tasks. Additionally, the framework incorporates non–code-switched (monolingual) data to improve robustness and generalization.

The following subsections describe the overall architecture and training strategies applied to CSDI.

3.1 Architecture

Figure 1 summarizes the flow of information through the model, from romanized input to the multitask outputs. CSDI is built around a shared multilingual transformer encoder that learns a unified representation for CSD, NER, and POS tagging. This shared design enables linguistic information learned by auxiliary tasks to support CSD, which is typically the lowest-resource component in Indic settings.

Shared Encoder

We experimented with IndicBERT, XLM-RoBERTa, mBERT, and MuRIL, and found MuRIL to offer the best performance on romanized Indic text, especially for languages with sparse training data. These results are discussed in more detail in Section 6.

Matrix-Language Conditioning

Although all input is romanized, the underlying Indic languages exhibit different syntactic and lexical tendencies, which influence switching behaviour. To encode this information, CSDI includes a trainable matrix-language embedding. The '+' operation in the architecture denotes element-wise addition. For each input sentence, the embedding corresponding to its base Indic language is added to the encoder's output. This conditioning shifts the internal representation to reflect language-specific tendencies—for example, the higher likelihood of English verb insertions in Hindi compared to Kannada.

Task-Specific Heads

Three lightweight linear classifiers operate on the conditioned encoder output (Figure 1):

- **CSD head**: predicts English vs. Indic token identity.
- **NER head**: predicts BIO entity types.
- **POS head**: predicts universal POS tags.

Each head performs word-level classification using the first token in the token sequence corresponding to each word. The auxiliary signals from POS and NER help the encoder distinguish structural and semantic contexts in which code-switching is more likely.

The multi-task setup encourages the encoder to learn syntactic boundaries (from POS) and semantic constraints (from NER) that correlate with switching events, reducing confusion in ambiguous romanized contexts. For example, patterns such as English nouns within Hindi noun phrases or English verbs following Hindi auxiliaries are reinforced across tasks.

3.2 Training Strategy

CSDI is optimized with a linearly weighted multi-task loss:

$$\mathcal{L} = w_{csd}\,\mathcal{L}_{csd} + w_{ner}\,\mathcal{L}_{ner} + w_{pos}\,\mathcal{L}_{pos}.$$

Here, $w_{csd} = w_{ner} = w_{pos} = 1$ in the main experiments (ablation studies vary these weights).

All three task heads are trained with token-level cross-entropy loss. The '+' operation at the output side of the architecture represents additive loss aggregation rather than a combination of task predictions. Each task head produces an independent token-level loss, which are summed (optionally weighted) up to form a single scalar objective. The combined loss is then backpropogated through the task heads and the shared encoder.

To prevent high-resource NER/POS tasks from dominating the low-resource CSD signal, we employ a *two-phase strategy*:

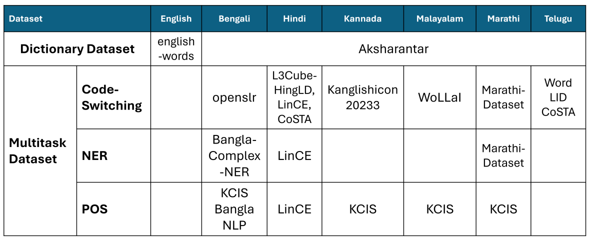
1. **Warm-up** (first epoch): the model is trained on a subset (≈20 %) of the CSD data with CSD loss only.

2. **Joint optimization** (remaining epochs): all three losses are combined.

This mitigates catastrophic forgetting while preserving the focus on CSD.

We conduct a set of ablation studies to understand the contribution of each component of CSDI. These include: (i) task ablations (CSD-only, CSD+POS, CSD+NER, full MTL), (ii) encoder selection (IndicBERT, mBERT, XLM-RoBERTa, MuRIL), (iii) data ablations (removing dictionary data, removing monolingual data, disabling romanization augmentation), and (iv) language-transfer settings (single-language models, family-grouped models, and all-language joint training). These ablations isolate the effect of auxiliary tasks, encoder choice, and cross-lingual transfer. POS and NER objectives serve only as auxiliary supervision during training; all reported results correspond to the CSD task.

The family-grouped and all-language models test the hypothesis that typologically related languages share switching patterns, enabling low-resource languages (e.g., Marathi) to benefit from high-resource members of the same family (e.g., Hindi). Future extensions could explore dynamic loss weighting for finer task balancing.



Dataset		English	Bengali	Hindi	Kannada	Malayalam	Marathi	Telugu
Dictionary Dataset		english-words	Aksharantar					
Multitask Dataset	**Code-Switching**		openslr	L3Cube-HingLD, LinCE, CoSTA	Kanglishicon 20233	WoLLal	Marathi-Dataset	Word LID CoSTA
	NER		Bangla-Complex-NER	LinCE			Marathi-Dataset	
	POS		KCIS Bangla NLP	LinCE	KCIS	KCIS	KCIS	

Figure 2: Overview of the datasets used for training CSDI. The table lists all dictionary, CSD, NER, and POS resources included for each Indic language. Complete references are listed in the Appendix.

4 Used Datasets

In this section, we describe the datasets used for training and evaluation. We make use of both publicly available and internal resources. The training data for CSDI contains a diverse collection of Indic–English code-switched data. All texts are transliterated into Latin script to eliminate orthographic variation and enable cross-language generalization. The training datasets include a dictionary-based dataset and a multitask dataset (for POS, NER, and CSD). Figure 2 summarizes the training data composition. A more detailed list of the used dataset can be found in Table 8.

4.1 Dictionary Train Dataset

This dataset consists of romanized word lists for multiple languages. For English, we use the *english-words* database[1], while for Indic languages we extract the top 10,000 most frequent words from the Aksharantar corpus (Madhani et al., 2023) and convert them into Latin script using indic-transliteration library[2]. These lists provide broad lexical coverage and help the model handle unseen romanized forms.

4.2 Multitask Train Dataset

The Multitask Dataset includes CSD, NER, and POS components. The NER and POS datasets are classical NLP datasets, while the CSD component consists of token-labeled sentences. Figure 3 provides an overview.

Most datasets are manually annotated, but for resources such as CoSTA, OpenSLR, and Kanglish, language labels are assigned automatically using a script heuristic: tokens in Latin script are tagged as English, and tokens in Indic scripts as the corresponding Indic language. This silver labeling introduces noise — for example, Indic Named Entities written in Latin script may be incorrectly tagged as English, while English-origin borrowed words (e.g., बस (bus) or ट्रेन (train)) written in Indic scripts may be mislabeled as Indic — but still provides valuable weak supervision.

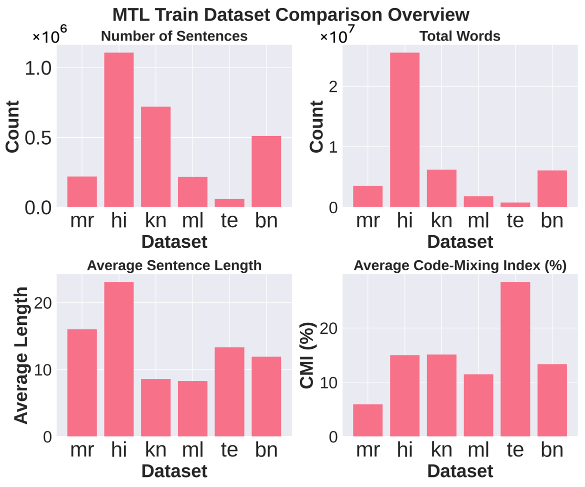


Figure 3: Overview of the MTL training data across six Indic languages. The figure summarizes the number of sentences, total words, average number of words in a sentence, and average CMI for each language.

As seen in Fig. 3, the Code-Mixing Index (CMI) varies across languages, reflecting domain differences. Marathi data, drawn mostly from formal text, contains fewer English insertions, while Telugu data, sourced largely from social media, shows more conversational code-switching.

4.3 Multitask Test Dataset

Although CSDI is trained in a multitask setting using CSD, POS, and NER supervision, evaluation is performed exclusively on the CSD task (token-level language identification). The General and Internal test sets therefore contain only token-level language identification labels. Accordingly, both the General and Internal test sets contain only token-level language identification labels too. The CSD part of the Multitask Test Dataset has the same composition as the Multitask Train Dataset, a mixture of gold- and silver-labeled data. All test data are held out and not used during training.

For each language pair, we reserve 1,000 sentences for the test split, except for Hindi, for which we use 6,000 held-out sentences. This results in a slightly uneven distribution across languages, but it reflects the availability of annotated

[1]https://github.com/dwyl/english-words

[2]https://pypi.org/project/ai4bharat-transliteration/

resources and ensures sufficient evaluation coverage for Hindi. Datasets that provide POS or NER annotations are used solely as auxiliary learning signals during training and are not evaluated at test time.

4.4 Pre-processing

Since the datasets originate from various domains and scripts, several normalization steps are applied prior to generating augmented romanized variants:

- removal of URLs, mentions, hashtags, emojis, numbers, and punctuation;
- explicit token-level language tagging: English tokens remain in Latin script, while Indic tokens are transliterated into their native script;
- treatment of intra-word code-switching (common in Malayalam) by assigning the embedded English label to the whole token;
- cleaning of the Aksharantar dictionary to remove English-origin words listed under Indic vocabularies.

The Aksharantar Indic dictionaries occasionally contain transliterated English-origin words (e.g., quarantine, audience) that are either direct borrowings or near-phonetic renderings in Indic script. Retaining such entries as Indic tokens can introduce systematic noise in code-switching detection, as these forms are likely to appear as English insertions in naturally mixed text.

To mitigate this, we identify and remove English-origin entries from the Indic dictionaries using a semi-automatic, overlap-based filtering procedure. Details of the filtering criteria and overlap analysis are provided in the Appendix.

These steps standardize the heterogeneous corpora and ensure consistent input before augmentation and training.

4.5 Augmented Dataset

The above datasets do not fully capture the variation introduced by informal romanization. To address this, we introduce an *Augmented Dataset* modeling cross-script and dialectal spelling variation. Romanization across speakers is highly inconsistent: for example, the English word *school* may appear as *school*, *skool*, or *iskul*, while Hindi किताब (book) may be written as *kitab* or *keetab*. As no resource exhaustively covers these forms, we generate variants automatically.

All training sentences are first transliterated into their native scripts and then back-transliterated into Latin script using three transliteration libraries:[3][4][5]. Because each library follows different conventions, this round-trip produces multiple plausible romanizations. To avoid excessive duplication, roughly 40% of tokens in each sentence are replaced with their variants, yielding up to five augmented versions per sentence. This improves robustness to spelling noise and non-standard romanization.

4.6 Internal Movie Dialogue Test Dataset

To evaluate performance on naturally occurring conversational speech, we use an *Internal Movie Dialogue Test Dataset* consisting of transcribed episodes from SonyYAY! [6] and SonyLIV [7]. Each episode is transcribed in the script of the audio track (e.g., Devanagari for Hindi), and the code-switched segments follow the same script.

SonyYAY! provides one animated episode each for Hindi, Marathi, and Telugu. SonyLIV provides one live-action episode each for Hindi, Malayalam, Marathi, and Telugu. Unlike the Multitask Test set, this dataset is smaller and reflects real dialogue rather than curated text.

Malayalam exhibits a lower CMI, partly due to annotator treatment of intra-word mixing: forms such as *drive-cheythu* or *call-um* were labeled as Malayalam, while during training such hybrids were treated as English based on their lexical roots. This mismatch contributes to lower Malayalam performance on this internal dataset.

5 Experiments

We evaluate CSDI on six Indic–English language pairs: Bn, Hi, Kn, Ml, Mr, and Te. All models are trained on romanized text with a weighted multitask loss and fine-tuned for 3 epochs using a learning rate of 1×10^{-5} and a batch size of 8. Performance is reported using macro F1, English F1 (the primary metric), and ΔCMI.

[3]https://indic-trans.readthedocs.io/
[4]https://pypi.org/project/ai4bharat-transliteration/
[5]https://github.com/indic-transliteration/indic_transliteration_py
[6]https://en.wikipedia.org/wiki/Sony_YAY!
[7]https://en.wikipedia.org/wiki/SonyLIV

5.1 Evaluation Setup

We use two held-out test sets: (i) a manually curated general test set, and (ii) a noisy internal movie-dialogue test set containing naturally occurring code-switching. These allow us to evaluate both clean, annotation-controlled performance and real-world robustness. We compare CSDI against AnE (Sterner, 2024), a recent multilingual CS detector based on XLM-RoBERTa.

Beyond token-level accuracy, it is important that a code-switching detector preserves the overall degree of mixing present in the input. We therefore report ΔCMI, defined as the difference between the predicted and gold CMI values (Section 2), to assess how faithfully a model reproduces the original mixing density.

A low absolute ΔCMI indicates that the model does not systematically over-predict or suppress English tokens, even when token-level errors occur. This property is particularly important for downstream tasks such as code-switched machine translation, where preserving the proportion of embedded language is often critical. In this sense, ΔCMI complements F1-based metrics by measuring distributional faithfulness rather than local accuracy alone.

5.2 Ablation Settings

To isolate the contribution of different components, we evaluate several model variants differing in task configuration, encoder choice, data composition, and cross-lingual training setup. The specific ablations and their results are presented in Section 6.

6 Results and Discussion

6.1 Main Results

The results in Table 1 correspond to the best-performing CSDI configuration, which uses the MuRIL encoder and non-uniform task weights (CSD: 2.0, POS: 1.5, NER: 1.0). This setup emerged as optimal in our ablation studies and is therefore used for all reported main results. Under this setup, CSDI consistently outperforms the AnE baseline (Sterner, 2024) across all six language pairs, with substantial improvements in English F1, macro F1, and ΔCMI. In contrast to AnE, however, who can work for any base language, our system is confined to those languages for which it was trained.

ΔCMI captures how faithfully a model reproduces the global degree of code-mixing in an utterance. CSDI obtains near-zero ΔCMI on the General test set, indicating that it not only identifies English tokens accurately but also models the overall mixing behaviour, an essential property for downstream MT systems that rely on retaining English insertions.

Since CSDI formulates detection as a binary task (English vs. L1), English F1 is the primary metric: English tokens are sparse yet carry most of the switching signal. CSDI shows the largest improvements for lower-resource languages such as Marathi and Bengali, demonstrating strong cross-lingual transfer from high-resource counterparts.

To assess robustness under noisier, real-world conditions, we additionally evaluate CSDI on the Internal Movie Dialogue test set. As expected, performance on this dataset is lower due to inconsistent automatic romanization and conversational style variation. Nevertheless, CSDI maintains strong performance across languages, showing reliable generalization despite domain mismatch. Full internal dataset results are provided in Table 13.

6.2 Encoder Selection

We compare four pretrained encoders—IndicBERT, mBERT, XLM-RoBERTa, and MuRIL—to determine which best models romanized Indic–English text. On the Internal test set (Table 2), MuRIL achieves the strongest overall performance, with higher English F1 and more stable ΔCMI than the other encoders; similar trends are observed on the General test set. This advantage likely stems from MuRIL's exposure to romanized Indic data during pretraining, which improves its handling of spelling variability and noisy transliterations. In contrast, mBERT performs weakest, and XLM-R shows reduced robustness to romanization noise. Based on these results, we use MuRIL as the encoder for all subsequent experiments.

6.3 Task Contribution

Table 3: Task contribution comparison on internal test data.

Task Combination	Macro F1	Prec. EN	Prec. L1	ΔCMI
CSD + POS	0.7731	0.5063	0.9711	-9.83
CSD + NER + POS	**0.8758**	**0.7300**	**0.9760**	**-2.65**
CSD + NER	0.7027	0.3977	0.9663	-16.09

Table 3 & 4 shows that auxiliary tasks substantially improve CSD performance, with POS providing

Table 1: Final results for all language pairs on the General test data. L1 denotes the base Indic language in each pair.

Lang. Pair	System	English			L1			Overall		
		P	R	F1	P	R	F1	Acc	MF1	ΔCMI
Hi–En	AnE	0.93	0.70	0.80	0.89	0.98	0.93	0.90	0.86	+7.26
	CSDI	0.98	0.96	**0.97**	0.98	0.99	0.99	0.98	**0.98**	**+0.48**
Mr–En	AnE	0.83	0.95	0.88	0.98	0.95	0.97	0.95	0.93	+2.65
	CSDI	0.93	0.94	**0.94**	0.98	0.98	0.98	0.98	**0.96**	**-0.70**
Bn–En	AnE	0.79	0.89	0.84	0.94	0.88	0.91	0.88	0.87	-4.33
	CSDI	0.95	0.95	**0.95**	0.97	0.97	0.97	0.97	**0.96**	**-0.29**
Kn–En	AnE	0.88	0.96	0.92	0.93	0.81	0.86	0.90	0.89	+5.12
	CSDI	0.92	0.95	**0.93**	0.93	0.87	0.90	0.92	**0.91**	**+2.40**
Ml–En	AnE	0.99	0.81	0.89	0.66	0.98	0.79	0.86	0.84	-13.30
	CSDI	0.99	0.99	**0.99**	0.99	0.99	0.99	0.99	**0.99**	**+0.08**
Te–En	AnE	0.93	0.93	0.93	0.92	0.92	0.92	0.93	0.93	-0.07
	CSDI	0.98	0.99	**0.99**	0.99	0.98	0.98	0.98	**0.98**	+0.32

Table 2: Encoder comparison on the internal dataset (English F1 only).

Encoder	**HI**	**MR**	**ML**	**TE**
IndicBERT	0.728	0.551	0.437	0.757
mBERT	0.691	0.467	0.443	0.763
MuRIL	**0.796**	**0.563**	0.245	**0.764**
XLM-R	0.715	0.521	**0.480**	0.762

Table 4: Task contribution comparison on General test performance for Hi-En.

Task Combination	Macro F1	Prec. EN	Prec. L1	ΔCMI
CSD + NER	0.9827	0.9836	0.9867	0.48
CSD + NER + POS	0.9747	0.9655	0.9847	**0.09**
CSD + POS	**0.9843**	0.9819	0.9893	0.24

the strongest gains. This likely stems from POS cues marking syntactic boundaries where switches frequently occur (e.g., Hindi noun before an English verb: "मैं गाना *play* करता हूँ"). In contrast, adding large amounts of non–code-switched NER data can reduce accuracy by biasing the model toward monolingual behavior. The full multitask configuration (CSD + NER + POS) achieves the best overall performance, demonstrating that combining syntactic and semantic signals is beneficial for modeling code-switching.

6.4 Data Contribution

Table 5 summarizes how different data sources affect model performance. Removing the dictionary hurts generalization to unseen romanized forms, and dictionary-only training leads to skewed predictions. Monolingual data provides small but consistent gains by improving contextual modeling. Augmented romanization variants, especially for English tokens in non-standard forms such as "*iskul*" for "*school*", improve recall on noisy internal data.

Table 5: Data contribution comparison on General test data.

Dataset Config	Macro F1	Prec. EN	Prec. L1	ΔCMI
All Data	**0.9723**	**0.9647**	**0.9823**	**+0.25**
Dictionary Data	0.2572	0.2897	0.7061	+25.33
CSD + Monolingual	0.2868	0.2851	0.6590	+20.50
Only CSD Data	0.4025	0.2938	0.7218	+3.42

Table 6: Impact of different training data components on Internal test performance for Hi-En.

Dataset Config	Macro F1	Prec. EN	Prec. L1	ΔCMI
All Data	**0.8717**	0.7102	0.9783	-3.41
Only Dictionary	0.1678	0.1193	0.6224	4.53
CSD + Monolingual	0.2392	0.1382	0.8157	-0.41
Only CSD	0.2803	0.1375	0.8269	-6.67

6.5 Language Grouping and Transfer Learning

To assess cross-lingual transfer, we compare three training strategies: (i) training each language independently, (ii) grouping languages by language family (Indo-Aryan or Dravidian), and (iii) training all six languages jointly. The impact of family-based grouping relative to individual training is summarized in Table 7, while the comparison between individual and fully joint training is reported in the supplementary material (Table 12).

Table 7: Effect of family-level grouping on the General test set. "Group-IA" = Indo-Aryan grouped; "Group-DR" = Dravidian grouped.

Language	Model Type	Macro F1	Prec. EN	Prec. L1
Hindi	Individual	**0.9723**	0.9647	0.9823
	Group - IA	0.9718	0.9642	0.9819
Bengali	Individual	0.8072	0.8154	0.8405
	Group - IA	**0.8243**	0.8235	0.8562
Marathi	Individual	0.6711	0.4194	0.9108
	Group - IA	**0.8875**	0.8723	0.9409
Telugu	Individual	**0.8586**	0.9020	0.8365
	Group - DR	0.7129	0.6216	0.9605
Kannada	Individual	0.5854	0.5652	0.6056
	Group - DR	**0.7920**	0.7734	0.8102
Malayalam	Individual	**0.7977**	0.8585	0.9688
	Group - DR	0.7677	0.8429	1.0000

Within-family grouping produces the strongest benefits, particularly for low-resource Marathi and Kannada. These improvements stem from shared English borrowing patterns and similar syntactic environments for mixing within the Indo-Aryan and Dravidian families. High-resource Hindi remains stable, showing only a minimal reduction due to mild negative transfer. Malayalam exhibits a small drop, which we attribute to inconsistent intra-word annotation conventions in the available training data.

We additionally evaluated parameter-efficient transfer through encoder freezing and adapter-based learning (results summarized in the supplemental material A.2). Freezing most of the encoder while fine-tuning only shallow task heads already yields competitive performance, especially for Marathi and Telugu, indicating that substantial cross-lingual transfer arises from shared subword representations. Adapter-based transfer further shows that lightweight modules trained jointly with multitask supervision recover a large portion of the full-model performance, making them attractive for scaling CSDI to new Indic languages with minimal additional parameters.

Together, these findings support strong within-family transfer and demonstrate that CSDI can be adapted effectively to new languages—either through full multitask fine-tuning or more parameter-efficient techniques.

6.6 Comparison with AnE

The final results show that CSDI matches or outperforms AnE across most language pairs in macro F1 while maintaining a smaller ΔCMI, with the largest gains on low-resource pairs (Mr–En, Bn–En). This indicates a more accurate estimation of code-mixing intensity, which is crucial for maintaining switch balance during translation.

A common weakness observed in the AnE system is that it often misclassifies homographic tokens between English and Indic in romanized form. For example, the Hindi phrase "के लिए" can appear as "*ke liye*" or "*key lie*", overlapping with English dictionary entries and leading AnE to falsely predict English. CSDI, by contrast, leverages contextual embeddings and multitask signals (NER, POS) to correctly disambiguate many such cases.

However, both models still struggle with context-dependent named entities. For example, in "*Honey, kal mat aana*" ("Honey, don't come tomorrow"), "*Honey*" is a person's name rather than the English noun, yet both models misclassify it as English due to limited context within a single sentence.

6.7 Language-wise Analysis

- **Hi:** CSDI attains near-perfect F1, largely due to the abundance of training data for this pair.
- **Mr:** Marathi shows the largest ΔCMI improvement, suggesting better modeling of code-switching boundaries despite limited data.
- **Bn, Kn, Te:** These languages show strong generalization with improved or comparable ΔCMI and stable macro F1, indicating robust behavior across both Indo-Aryan and Dravidian families.
- **Ml:** CSDI performs poorly on the Malayalam–English pair on the internal dataset but nearly perfectly on the General dataset. This discrepancy is driven by inconsistent intra-word labeling: annotators labeled English-origin stems (e.g., "*drive-cheythu*", "*call-um*") as Malayalam, whereas our training scheme treated English lexical roots as English. This mismatch inflates false negatives and highlights the need for standardized intra-word code-switching annotation guidelines.

7 Conclusion

CSDI introduces a multitask learning framework for Indic–English code-switching detection that jointly models CSD, NER, and POS through a shared encoder with matrix-language conditioning.

Across six Indic language pairs, the model delivers strong and consistent gains, achieving state-of-the-art macro F1 and near-zero ΔCMI, and outperforming the AnE baseline in both accuracy and code-mixing density estimation.

The effectiveness of CSDI stems from its ability to exploit syntactic and semantic cues from auxiliary tasks—particularly POS—to identify linguistically plausible switching boundaries. This architecture also enables meaningful cross-lingual transfer: high-resource languages such as Hindi provide structural signals that substantially benefit lower-resource languages like Marathi. The model's stability across both clean and noisy evaluation settings underscores its practical utility for token-level language identification in code-mixed Indic text.

Overall, CSDI offers a robust and extensible foundation for scalable multilingual code-switching detection and serves as a reliable component for downstream applications, including code-switched machine translation.

Limitations

CSDI, in its current form, handles only full-word code-switching. Intra-word mixing is assigned a single label based on the lexical origin of the token, which simplifies annotation but prevents the model from capturing finer grained subword structure — an issue particularly relevant for languages such as Malayalam. Addressing this would require tokenizers or encoders operating at character or byte levels.

The model also relies on sentence-level context and therefore struggles with discourse-dependent cases, including person names and contextually grounded entities that cannot be disambiguated from local information alone. Similarly, Romanized spellings that differ substantially from the training distribution reduce recall for English tokens, a consequence of inconsistent orthography and social-media spelling variation.

CSDI further assumes a single embedded language (English) within a base Indic language. While this reflects the predominant pattern in our datasets, it does not capture multilingual settings where more than one embedded language may appear.

Finally, the approach benefits from auxiliary supervision via NER, POS, and dictionary resources. For extremely low-resource languages where such datasets are limited or unavailable, the gains from multitask learning may diminish, suggesting the need for task-agnostic or unsupervised alternatives.

References

Gustavo Aguilar, Sudipta Kar, and Thamar Solorio. 2020. Lince: A centralized benchmark for linguistic code-switching evaluation. *Preprint*, arXiv:2005.04322.

A. Bolonyai. 2009. Code-switching, imperfect acquisition, and attrition. In Barbara E. Bullock and Almeida Jacqueline Toribio, editors, *The Cambridge Handbook of Linguistic Code-switching*, Cambridge Handbooks in Language and Linguistics, pages 253–269. Cambridge University Press, Cambridge.

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

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).

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

François Grosjean. 2010. *Bilingual: Life and Reality*. Harvard University Press.

Janice Jake and Carol Myers-Scotton. 1997. Codeswitching and compromise strategies: Implications for lexical structure. *International Journal of Bilingualism - INT J BILING*, 1:25–39.

Anupam Jamatia, Amitava Das, and Björn Gambäck. 2018. Deep learning-based language identification in english-hindi-bengali code-mixed social media corpora. *Journal of Intelligent Systems*, 28.

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 the Association for Computational Linguistics: EMNLP 2020*, pages 4948–4961. Association for Computational Linguistics.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip

Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages. Preprint, arXiv:2103.10730.

Yash Madhani, Sushane Parthan, Priyanka Bedekar, Gokul Nc, Ruchi Khapra, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh Khapra. 2023. Aksharantar: Open Indic-language transliteration datasets and models for the next billion users. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 40–57, Singapore. Association for Computational Linguistics.

Pieter Muysken. 1995. *Code-switching and grammatical theory*, pages 177–198. Cambridge University Press, Cambridge.

Parth Patwa, Gustavo Aguilar, Sudipta Kar, Suraj Pandey, Srinivas PYKL, Björn Gambäck, Tanmoy Chakraborty, Thamar Solorio, and Amitava Das. 2020. SemEval-2020 task 9: Overview of sentiment analysis of code-mixed tweets. In *Proceedings of the Fourteenth Workshop on Semantic Evaluation*, pages 774–790, Barcelona (online). International Committee for Computational Linguistics.

Omkar Pitale and Aruna Malapati. 2023. Leveraging multi-task learning in code-switched setting. *Preprint*.

Shana Poplack. 1981. Syntactic structure and social function of code-switching. In Richard P. Durán, editor, *Latino Language and Communicative Behavior*, pages 169–184. Ablex, Norwood, NJ.

Adithya Pratapa and Monojit Choudhury. 2017. Quantitative characterization of code switching patterns in complex multi-party conversations: A case study on Hindi movie scripts. In *Proceedings of the 14th International Conference on Natural Language Processing (ICON-2017)*, pages 75–84, Kolkata, India. NLP Association of India.

Sebastin Santy, Anirudh Srinivasan, and Monojit Choudhury. 2021. BERTologiCoMix: How does code-mixing interact with multilingual BERT? In *Proceedings of the Second Workshop on Domain Adaptation for NLP*, pages 111–121, Kyiv, Ukraine. Association for Computational Linguistics.

HAZ Sameen Shahgir, Ramisa Alam, and Md. Zarif Ul Alam. 2023. Banglaconer: Towards robust bangla complex named entity recognition. *Preprint*, arXiv:2303.09306.

Igor Sterner. 2024. Multilingual identification of English code-switching. In *Proceedings of the Eleventh Workshop on NLP for Similar Languages, Varieties, and Dialects (VarDial 2024)*, pages 163–173, Mexico City, Mexico. Association for Computational Linguistics.

Genta Indra Winata, Andrea Madotto, Chien-Sheng Wu, and Pascale Fung. 2018. Code-switching language modeling using syntax-aware multi-task learning. In *Proceedings of the Third Workshop on Computational Approaches to Linguistic Code-Switching*, pages 62–67, Melbourne, Australia. Association for Computational Linguistics.

A Appendix

A.1 Dataset Supplementary Material

A.1.1 Training and Test Set Overviews

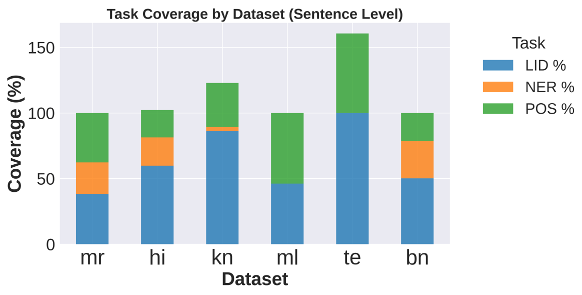


Figure 4: Task coverage across training datasets, showing the proportion of sentences annotated with CSD (LID), NER, and POS for each language.

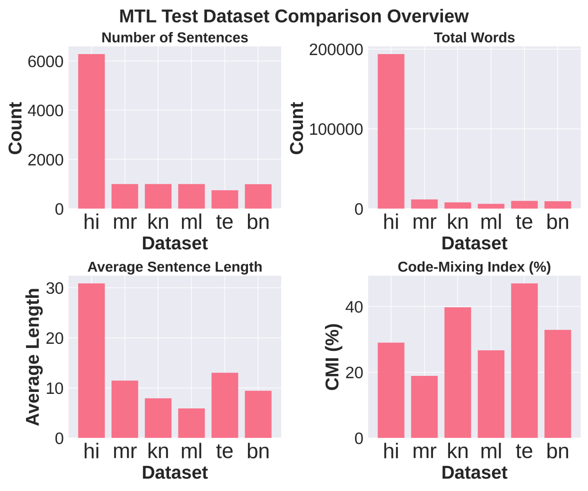


Figure 5: General Test Dataset statistics: number of sentences, total word count, average sentence length, and CMI for each language.

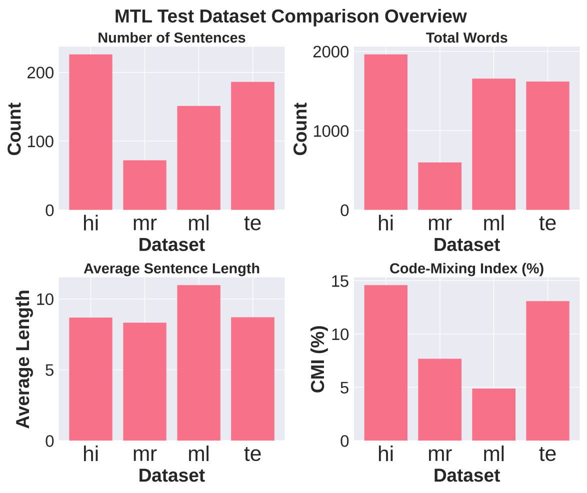


Figure 6: Internal Test Dataset statistics: number of sentences, total words, average sentence length, and CMI across languages.

A.1.2 Dataset Inventory

Table 8 provides a consolidated inventory of all datasets used in this work.

Table 8: Dataset used with descriptions and sources.

Dataset	Description	Source
english-words	Curated English word list used as a lexical resource	https://github.com/dwyl/english-words
Aksharantar	Indic transliteration corpus	(Madhani et al., 2023)
OpenSLR	OpenSLR-104 code-switched Indic transcripts	https://www.openslr.org/104/
L3Cube-HingLID	Hindi–English code-mixed language identification dataset	https://github.com/l3cube-pune/code-mixed-nlp
Kanglishicon	Kannada–English code-mixed text dataset	https://sites.google.com/view/kanglishicon2022/dataset
WoLLaI	WoLLaI Mal-Eng	https://data.mendeley.com/datasets/tzrcrrwz4n/1
L3Cube-MahaNLP	Marathi dataset subset used in prior code-mixed NLP releases	https://github.com/Harry262000/Marathi-Dataset
CoSTA	Silver labelled dataset for Bn, Hi, Mr, Te	https://github.com/csalt-research/CoSTA
Bangla-Complex-NER	Bangla complex named entity recognition dataset	(Shahgir et al., 2023)
LINCE	LINCE benchmark providing Hindi NER and POS annotations	(Aguilar et al., 2020)
KCIS (Bn, Kn, Mr, Ml)	POS-tagged Indian language resources released by IIIT-Hyderabad	https://ltrc.iiit.ac.in/showfile.php?filename=downloads/kolhi/
Telugu-POS	Telugu–English POS-tagged code-mixed dataset	GitHub: SunilGundapu / Word-Level-LID-TE

A.1.3 Filtering English-Origin Entries in Indic Dictionaries

Aksharantar Indic dictionaries contain a small but non-negligible number of English-origin words transliterated into Indic scripts. While such forms may be considered lexical borrowings, treating them as Indic tokens introduces ambiguity for token-level code-switching detection, where the goal is to identify English insertions in Indic text.

To quantify this overlap, we compare transliterated dictionary vocabularies across languages against an English word list and computed lexical overlap statistics. We observe a non-trivial overlap between English and Indic dictionaries, as well as homographic collisions across Indic languages themselves. Figure 7 illustrates the pairwise Jaccard similarity of dictionary vocabularies across languages and English.

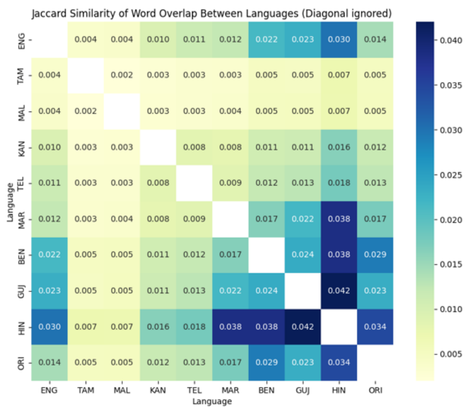


Figure 7: Pairwise Jaccard similarity of word overlap between transliterated dictionary vocabularies across languages and English (diagonal ignored). Higher overlap between English and Indo-Aryan languages reflects a greater presence of shared or borrowed romanized forms.

We observe substantially higher overlap between English and Indo-Aryan languages (e.g., Hindi, Marathi, Bengali, Gujarati) than with Dravidian languages. This pattern motivates more aggressive filtering of English-origin entries for Indo-Aryan dictionaries in order to reduce systematic noise during training.

Based on this analysis, dictionary entries with high-confidence overlap with English forms were removed from the Indic dictionaries prior to training. Table 9 shows representative examples of English-origin entries removed from the Hindi dictionary; similar patterns are observed for other languages.

Table 9: Examples of English-origin entries removed from the Hindi dictionary.

Indic form	Romanized form	English gloss
क्रैडल	cradle	cradle
क्वॉरेनटाइन	quarantine	quarantine
इअर	ear	ear
ऑडिएंस	audience	audience
ऐंजल	angel	angel
लूवे	loove	love
ऑफर:	offerh	offer

A.1.4 Example of Romanization Augmentation

To illustrate the effect of our round-trip transliteration augmentation, consider the Hindi sentence:

kal school mai new teacher aai thi

Augmentation yields several plausible variants:

- *kal skool mai new techar ayi thi*
- *kal iskool me new teacher aayi thi*
- *kal school mey nyu teacher aaai thi*
- *kal skul mai new teacher ayi thi*
- *kal skool me new ticher aayee thi*

These reflect natural variation found in social media and informal writing.

A.2 Transfer Learning

A.2.1 Encoder Freezing Experiments

Table 10 reports performance on the Internal test set for multiple encoder-freezing strategies compared against full fine-tuning during sequential transfer from Hindi to Marathi and from Kannada to Telugu and Malayalam. The evaluated variants include partial freezing of different encoder segments (Low–Mid: layers 1–6, Mid–High: layers 6–12, and Mid: layers 4–9), freezing the entire encoder ("Encoder"), and freezing both the encoder and the POS head while training only the remaining task heads ("Encoder+POS"). Across all languages, freezing the full encoder consistently outperforms both partial freezing strategies and full fine-tuning.

Table 10: Performance of partial encoder freezing vs. full finetuning for Mr, Te, Ml on internal test data

Lang	Low–Mid	Mid–High	Mid	Encoder	Enc+POS	Full FT
MR	0.609	0.500	0.566	**0.660**	0.604	0.569
TE	0.794	0.807	0.776	**0.811**	0.806	0.803
ML	0.228	0.494	0.213	**0.506**	0.447	0.456

A.2.2 Adapter-Based Transfer

Table 11 presents a comparison between adapter-based transfer strategies and full multitask fine-tuning for Marathi and Telugu on Internal test set. Language-specific adapters trained in isolation perform poorly, while zero-shot transfer using a Hindi-trained adapter provides a clear improvement. Combining language adapters with multitask learning substantially improves performance, narrowing the gap to full multitask fine-tuning. Overall, the results indicate that adapters offer a viable parameter-efficient alternative, although full fine-tuning remains the strongest setting.

Table 11: Adapter-based transfer vs. full multitask fine-tuning for Mr, Te.

Lang	Lang Adapter	Lang+MTL	Zero-shot	Full MTL
MR	0.3418	0.5208	0.4951	**0.6078**
TE	0.5737	0.7921	0.7625	**0.7915**

A.2.3 Six-Language Joint Training

Earlier, we compared individual-language training with family-based grouping of related languages. Here, we extend this analysis by comparing individual-language models with joint training across all six languages.

Table 12: Comparison of individual-language models and joint six-language multitask training on the General test set.

Language	Model Type	Macro F1	Prec. EN	Prec. L1
Hindi	Individual	0.9723	0.9647	0.9823
	Joint	0.9721	0.9655	0.9818
Bengali	Individual	0.8072	0.8154	0.8405
	Joint	0.8056	0.7857	0.8481
Marathi	Individual	0.6711	0.4194	0.9108
	Joint	**0.8541**	0.7143	0.9572
Telugu	Individual	0.8586	0.9020	0.8365
	Joint	**0.9227**	0.9153	0.9301
Kannada	Individual	0.5854	0.5652	0.6056
	Joint	**0.8226**	0.7826	0.8661
Malayalam	Individual	0.7977	0.8585	0.9688
	Joint	**0.8112**	0.8634	1.0000

A.3 Full Results for All Test Sets

Table 13: Results on both the General test set and the Internal Movie Dialogue test set.

Lang. Pair	System	English			L1			Overall		
		P	R	F1	P	R	F1	Acc	MF1	$\lvert\Delta\text{CMI}\rvert$
					General Test Set					
Hi–En	AnE	0.93	0.70	0.80	0.89	0.98	0.93	0.90	0.86	7.26
	CSDI	**0.98**	0.96	**0.97**	0.98	0.99	0.99	0.98	**0.98**	**0.48**
Mr–En	AnE	0.83	0.95	0.88	0.98	0.95	0.97	0.95	0.93	2.65
	CSDI	**0.93**	0.94	**0.94**	0.98	0.98	0.98	0.98	**0.96**	0.70
Bn–En	AnE	0.79	0.89	0.84	0.94	0.88	0.91	0.88	0.87	4.33
	CSDI	**0.95**	0.95	**0.95**	0.97	0.97	0.97	0.97	**0.96**	**0.29**
Kn–En	AnE	0.88	0.96	0.92	0.93	0.81	0.86	0.90	0.89	5.12
	CSDI	**0.92**	0.95	**0.93**	0.93	0.87	0.90	0.92	**0.91**	**2.40**
Ml–En	AnE	0.99	0.81	0.89	0.66	0.98	0.79	0.86	0.84	13.30
	CSDI	0.99	0.99	**0.99**	0.99	0.99	0.99	0.99	**0.99**	**0.08**
Te–En	AnE	0.93	0.93	0.93	0.92	0.92	0.92	0.93	0.93	0.07
	CSDI	**0.98**	0.99	**0.99**	0.99	0.98	0.98	0.98	**0.98**	0.32
					Internal Movie Dialogue Test Set					
Hi–En	AnE	0.54	0.83	0.66	0.96	0.88	0.92	0.87	0.79	7.54
	CSDI	**0.74**	0.88	**0.80**	0.98	0.95	0.96	0.94	**0.88**	**2.65**
Mr–En	AnE	0.49	0.67	0.56	0.97	0.94	0.95	0.92	0.76	2.83
	CSDI	**0.60**	0.74	**0.66**	0.96	0.98	0.97	0.94	**0.81**	**0.00**
Ml–En	AnE	0.90	0.67	0.79	0.97	0.99	0.98	0.97	0.87	1.95
	CSDI	0.43	0.81	0.56	0.98	0.92	0.95	0.91	0.76	6.56
Te–En	AnE	0.75	0.82	0.78	0.97	0.96	0.96	0.94	0.87	1.05
	CSDI	**0.80**	0.79	**0.79**	0.97	0.97	0.97	0.95	**0.88**	**0.24**