ILID: Native Script Language Identification for Indian Languages

Yash Ingle, Pruthwik Mishra
Sardar Vallabhbhai National Institute of Technology, Surat, India
yash.ingle003@gmail.com,pruthwikmishra@aid.svnit.ac.in

Abstract

The language identification task is a crucial fundamental step in NLP. Often it serves as a preprocessing step for widely used NLP applications such as multilingual machine translation, information retrieval, question and answering, and text summarization. The core challenge of language identification lies in distinguishing languages in noisy, short, and code-mixed environments. This becomes even harder in case of diverse Indian languages that exhibit lexical and phonetic similarities, but have distinct differences. Many Indian languages share the same script, making the task even more challenging. Taking all these challenges into account, we develop and release a dataset of 250K sentences consisting of 23 languages including English and all 22 official Indian languages labeled with their language identifiers, where data in most languages are newly created. We also develop and release baseline models using state-of-the-art approaches in machine learning and fine-tuning pre-trained transformer models. Our models outperforms the state-of-the-art pre-trained transformer models for the language identification task. The dataset and the codes are available at https://yashingle-ai.github.io/ILID/ and in Huggingface open source libraries.

1 Introduction

India is a linguistically diverse country. Although there are more than 1000 languages in the Indian subcontinent, the digital divide among many Indian languages is enormous. Barring a few languages, most of the languages suffer from resource scarcity and a simple task such as Language IDentification (LID) remains a challenging task (Caswell et al., 2020) for them. Available LID tools such as Lui and Baldwin (2011), Joulin et al. (2016a), and Team et al. (2022) perform poorly on Indian languages and they do not cover many of them. With the increase in mobile and internet users in India, the need for providing services in native Indian languages is more pressing than ever. Hence, it becomes essential to create robust LID tools to cater to Indian users. As the volume and variety of data continue to grow, we need to include data from different domains while creating sound benchmarks. Keeping this motivation in mind, we set out to create an Indian Language Identification (ILID) benchmark including English and 22 official languages spanning 25 scripts consisting of a total of 250K sentences labeled with their language markers. The main contribution of our paper is two fold.

- We create the ILID dataset consisting of newly created datasets in 13 languages and curated datasets in the remaining ten languages.
- We develop ILID baseline models using different machine learning and deep learning techniques.
- We perform a comparative analysis of our models with the state-of-the-art model.

2 Related Work

Language Identification (LID) is a fundamental task in Natural Language Processing (NLP), aiming to determine the language of a given text or speech segment. Early approaches to LID primarily relied on statistical methods, most notably character n-gram models (Cavnar and Trenkle, 1994). These methods leverage the unique frequency distributions of character sequences within different languages to classify unknown texts. Language Identification becomes more crucial in the domain of Cross Lingual Information Retrieval where the query is written in native scripts (Bosca and Dini, 2010). Similarly, spoken language identification(Dehak et al., 2011; Gonzalez-Dominguez et al., 2014) is a very essential task in multilingual speech processing that can further impact in developing applications such as machine translation and speech recognition. In this paper, we are

dealing with the language identification task in written texts. With machine learning approaches such as Support Vector Machines (SVMs) (Dunning, 1994), Naive Bayes classifiers (Elworthy, 1998), decision trees, random forests, and gradient boosting gaining widespread usage, LID systems were also modeled to leverage these techniques with the same n-gram features and other lexical or language specific hand crafted features. While effective for well-resourced languages with distinct orthographies, the performance of the statistical systems can degrade when dealing with short texts, noisy data, or languages sharing common scripts. Deep learning methods such as Convolutional Neural Networks (CNNs) (Kim, 2014) and Recurrent Neural Networks (RNNs) (Hochreiter and Schmidhuber, 1997) (particularly LSTMs and GRUs) are often trained on distributionally similar word or character n-gram or single character embeddings vectors to better capture intricate language complexity and handle out-of-vocabulary words more robustly (Conneau et al., 2017). The current approaches include fine-tuning pre-trained transformer (Vaswani et al., 2017) models using subwords (Sennrich et al., 2016; Kudo and Richardson, 2018; Song et al., 2021). Many of the language identification tasks are modeled as token classification tasks that are helpful in code-mixed settings.

The task of language identification for Indian languages is even more challenging as they belong to different and diverse language families (Indo-Aryan, Dravidian, Tibeto-Burman, Austroasiatic), and written in over a dozen distinct scripts. Multiple Indian languages often share the same script. For example; Hindi, Marathi, Nepali, and Sanskrit are all primarily written in the Devanagari script. This script overlap requires models to distinguish languages based on lexical, morphological, or syntactic features rather than solely on orthographic cues.

Another prevalent issue in Indian language contexts is code-mixing and code-switching, where speakers or writers regularly use words or phrases from multiple languages within a single utterance or text (Gambäck and Das, 2017). This phenomenon is common in informal communication and social media (Barman et al., 2014), making it difficult for traditional LID systems to accurately identify the primary language or even segment code-mixed segments. Research in this area often focuses on fine-grained LID, aiming to identify language at the word or phrase level (Bhat et al., 2014), rather than just document or sentence level identification.

The language identification task for Indian languages also followed the similar line of research using rule-based and machine learning based approaches. More advanced approaches fine-tune pre-train transformers trained on Indian languages (Kumar et al., 2023; Agarwal et al., 2023; Madhani et al., 2023a). Madhani et al. (2023a) also transliterate (Madhani et al., 2023b) the native scripts into roman scripts for uniformity across languages and better efficiency in the LID systems. Despite these advancements, robust and fine-grained LID for all Indian languages, especially in code-mixed scenarios and for low-resource languages, remains an active area of research. In this paper, we limit the scope of identifying the language at a sentence level, not at the constituent token level.

3 ILID Dataset Creation

Indian Language Identification (ILID) dataset is created using two approaches. We include English and the 22 official Indian languages [1][2] widely used in India. The first approach utilizes web scraping for the languages in which the digital presence is significant (details are given in the Appendix). For each of these languages, we collect 10,000 different sentences from various sources such as government websites, newspapers, books, and other public materials, ensuring varying degrees of linguistic complexity. The second approach samples sentences from Bhashaverse (Mujadia and Sharma, 2024), an existing massive monolingual and parallel corpora for Indian languages. The details of the dataset are presented in Table 1. The data in each language is split into 80:10:10 ratio to create train, dev, and test sets. The dataset is available at https://huggingface.co/datasets/yash-ingle/ILID_Indian_Language_Identification_Dataset.

To ensure quality and consistency, the dataset undergoes several noise removal steps. The first step involves the elimination of duplicate, very short, and ungrammatical sentences. The next step employs an existing FastText (Joulin et al., 2016b,a) based language identification model to remove sentences where the probability of detecting the language of the text is very low (we fix a threshold of

[1] https://en.wikipedia.org/wiki/Eighth_Schedule_to_the_Constitution_of_India

[2] https://en.wikipedia.org/wiki/Linguistic_Survey_of_India

Language	#Train	#Dev	#Test	#Total
Assamese (asm)	8000	1000	1000	10000
Bengali (ben)	8000	1000	1000	10000
Bodo (brx)	8000	1000	1000	10000
Dogri (doi)	8000	1000	1000	10000
Konkani (gom)	8000	1000	1000	10000
Gujarati (guj)	8000	1000	1000	10000
Hindi (hin)	8000	1000	1000	10000
Kannada (kan)	8000	1000	1000	10000
Kashmiri (kas)	8000	1000	1000	10000
Maithili (mai)	8000	1000	1000	10000
Malayalam (mal)	8000	1000	1000	10000
Marathi (mar)	8000	1000	1000	10000
Manipuri Bengali Script (mni_Beng)	8000	1000	1000	10000
Manipuri Meitei Script (mni_Mtei)	8000	1000	1000	10000
Nepali (npi)	8000	1000	1000	10000
Odia (ory)	8000	1000	1000	10000
Punjabi (pan)	8000	1000	1000	10000
sanskrit(san)	8000	1000	1000	10000
Santali (sat)	8000	1000	1000	10000
Sindhi Devnagari(snd_Deva)	8000	1000	1000	10000
Sindhi Perso-Arabic(snd_Arab)	8000	1000	1000	10000
Tamil (tam)	8000	1000	1000	10000
Telugu (tel)	8000	1000	1000	10000
Urdu (urd)	8000	1000	1000	10000
English (eng)	8000	1000	1000	10000
Total	**200000**	**25000**	**25000**	**250000**

Table 1: Data splits for ILID Benchmark Dataset

0.7 for this filtering process). This happens in code-mixed texts as many Indians are multilingual and often use more than one language while writing.

The approaches are described in detail.

3.1 Web Scraping

To build a strong language identification system for Indian languages, we carefully built a personalized set of 10,000 text samples for each language, covering 13 Indian languages.

Text data is collected from a variety of publicly available and diverse sources, including Wikipedia dumps in respective languages for formal and structured text, news websites and blogs for professional and personal updates. All efforts are made to ensure the samples are multilingual in length, script (i.e., Devanagari, Bengali, Tamil, Odia etc.), domain, and style, reflecting the multilingual nature of India.

For a target website W, we define the scraping process as:

$$S(W) = \bigcup_{p \in P_W} \phi(p) \tag{1}$$

where P_W is the set of target pages and ϕ is the extraction function that parses HTML content while preserving structural information. We implement adaptive throttling using:

$$\Delta_t = \frac{1}{|W|} \sum_{i=1}^{|W|} \frac{\text{size}(w_i)}{\text{bandwidth}} \tag{2}$$

3.2 Data Cleaning Pipeline

In the data cleaning process, we mainly work on removing extra spaces, extra special symbols, unicode normalization, breaking the paragraph into meaningful sentences, and after that we apply various tokenization methods. For sentence tokenization, we utilize the corresponding end sentence markers defined for respective languages. We use

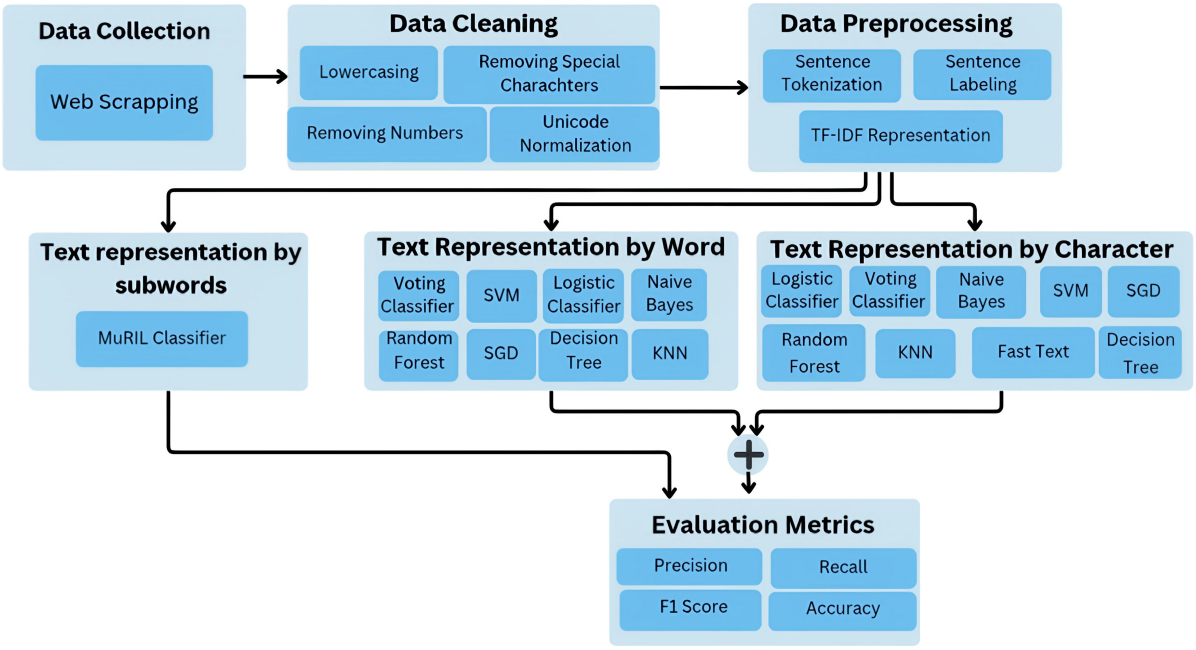

Figure 1: Workflow of the Proposed Language Identification System.

a regular expression based tokenizer [3] for sentence and word tokenization that is specifically designed to handle texts in Indian languages. Our cleaning process transforms raw text T to cleaned text T' through:

$$T' = \psi_n(\psi_s(T)) \quad (3)$$

where ψ_s handles special characters and ψ_n performs normalization. We use a sentence as a unit for the LID task. Each sentence is labeled with a language tag. ISO 639-2 codes [4] or 3 lettered language codes denote each language detailed in Table 3.

3.3 Sampling From Existing Corpora

In the second approach, sentences are randomly sampled from Bhashaverse (Mujadia and Sharma, 2024) that contain a huge collection of monolingual corpora of different Indian languages. In this corpora, Manipuri, and Sindhi have sentences in two different scripts each. Manipuri is available in Bengali and Meitei scripts, whereas Sindhi is written in Devanagari and Perso-Arabic scripts. We have sampled sentences using all available scripts from these two languages. This increases the total number of languages with different scripts to 25 also presented in Table 4. A similar exercise of appending the labels to the sentences as language identifiers and data splitting is performed on these samples. The language specifics using both approaches are shown in Table 2.

Scraping	Sampling
Assamese	Bodo
Bengali	Dogri
Gujarati	Kashmiri
Hindi	Konkani
Kannada	Maithili
Malayalam	Manipuri
Marathi	Nepali
Oriya	Sanskrit
Punjabi	Santhali
Tamil	Sindhi
Telugu	
Urdu	
English	

Table 2: Languages Used in Scraping and Sampling

3.4 Corpora Statistics

After collecting and cleaning the dataset, we compute several corpora statistics. These statistics are excellent indicators of the levels and complexities of each language. The properties of the corpora considered include the total number of words, the total number of characters, the total number of unique words, average sentence length, the average word length, and the type / token ratio (textinspector, 2025) which are markers of the representativeness of the corpora. These statistics are shown in

[3] https://github.com/Pruthwik/Tokenizer_for_Indian_Languages

[4] https://en.wikipedia.org/wiki/List_of_ISO_639-2_codes

Language	ISO 639-2 Code
Assamese	asm
Bangla	ben
Bodo	brx
Dogri	doi
Gujarati	guj
Hindi	hin
Kannada	kan
Kashmiri	kas
Konkani	gom
Maithili	mai
Malayalam	mal
Manipuri Bengali Script	mni_Beng
Manipuri Meitei Scrip	mni_Mtei
Marathi	mar
Nepali	npi
Oriya	ory
Punjabi	pan
Sanskrit	san
Santali	sat
Sindhi Perso-Arabic	snd_Arab
Sindhi Devanagari	snd_Deva
Tamil	tam
Telugu	tel
Urdu	urd
English	eng

Table 3: Language Full Name to ISO 639-2 Language Codes Mapping

Table 4.

4 ILID Model

The ILID model is a classifier that can categorize a piece of text into one of the 25 classes that includes English and 22 Indian languages. The architecture of the proposed system is shown in Figure 1. We implement three types of approach for designing the classifiers.

4.1 Machine Learning Models

In this approach, each sentence is represented by a TF-IDF (Sparck Jones, 1972) vector in a bag-of-words setting. We utilize both word-level and character-level TF-IDF representations to enhance textual feature extraction.

4.1.1 Word-Level TF-IDF Based Classification

We begin by utilizing word-level TF-IDF, computed using uni-grams and bi-grams. Higher word n-grams($n > 2$) are not used for modeling TF-IDF as they suffer from sparsity. This representation captures the importance of whole words and adjacent word pairs across the corpus, allowing the model to understand the semantic context and overall meaning of sentences. However, word-based models are sensitive to spelling variations and may not generalize well to morphologically rich or noisy text, which is common in Indian languages.

4.1.2 Character-Level TF-IDF Based Classification

To overcome the limitations of word-level models, we introduce character-level TF-IDF representations using character n-grams ranging from 2 to 6. This approach captures sub-word structures, prefixes, suffixes, and root forms that are particularly effective in handling typographical errors, morphological variations, and out-of-vocabulary words. Character-level features are especially helpful in language identification tasks involving noisy or informal text, enabling the model to learn finer linguistic patterns.

4.1.3 Combined Feature-Based Classification

To leverage the strengths of both word and character level features, we develop a combined TF-IDF representation by concatenating the two feature sets into a single vector. This hybrid approach provides a more robust and language-agnostic representation. We train eight traditional machine learning classifiers—Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RF), Decision Tree (DT), Naive Bayes (NB), Stochastic Gradient Descent (SGD), and K-Nearest Neighbors (KNN)—on these features. Furthermore, we explore various ensemble models involving combinations of 3, 4, and 5 diverse base classifiers from the pool of classifiers. Due to space constraints, only the top five performing ensembles are selected based on F1 scores on the development and test sets, are presented in Table 5.

4.2 FastText Classifier

FastText (Joulin et al., 2016b,a) utilizes word embeddings (Bojanowski et al., 2017) composed of character n-grams that better represent rare words and orthographic similarities. FastText classifier is a linear classifier and is very fast in computation. It can generate embeddings for out-of-vocabulary (OOV) words, which are common in user-generated or noisy text data.

lang	#sents	#words	#chars	avg_word_len	avg_sent_len	#unique_words	TTR_words
mai	10000	65884	288223	3.526	28.822	6743	0.102
eng	10000	225036	1052283	3.721	105.228	22709	0.101
kan	10000	122400	861095	6.117	86.109	34644	0.283
san	10000	130353	858829	5.665	85.883	50506	0.387
gom	10000	58282	296807	4.264	29.681	8104	0.139
hin	10000	106402	476919	3.576	47.692	8451	0.079
mar	10000	89232	526021	5.007	52.602	19470	0.218
asm	10000	64251	311313	4.001	31.131	7196	0.112
urd	10000	97233	387031	3.083	38.703	19987	0.206
tel	10000	81281	524216	5.572	52.422	19554	0.241
tam	10000	144307	1018268	6.126	101.827	45912	0.318
snd_Arab	10000	113249	479446	3.322	47.945	7409	0.065
ben	10000	170048	877189	4.217	87.719	30007	0.176
snd_Deva	10000	70743	312570	3.56	31.257	6249	0.088
ory	10000	91919	526927	4.841	52.693	5400	0.059
npi	10000	59095	296473	4.186	29.647	7966	0.135
mni_Mtei	10000	73043	318189	3.493	31.819	7245	0.099
mal	10000	117447	985898	7.48	98.59	37010	0.315
doi	10000	75469	313613	3.288	31.361	5904	0.078
pan	10000	168450	721285	3.341	72.129	18032	0.107
brx	10000	65717	343731	4.383	34.373	7828	0.119
sat	10000	78554	360814	3.72	36.081	4067	0.052
mni_Beng	10000	186109	1208020	5.545	120.802	27350	0.147
guj	10000	175966	911173	4.235	91.117	28195	0.16
kas	10000	69270	297326	3.437	29.733	7752	0.112

Table 4: Language Wise Corpus Statistics Where the column TTR_words refers to the Type Token Ratio for Words

4.3 Pretrained BERT Model

Pretrained subword based contextual language models such as BERT (Devlin et al., 2019), XLM (Conneau et al., 2020), RoBERTa (Zhuang et al., 2021) have proven to be very effective in text classification tasks or generally natural language understanding tasks. One variant of BERT, MuRIL (Khanuja et al., 2021) is pretrained on large amounts of corpora in Indian languages. The pretraining data is also augmented with translated and transliterated texts that makes the model capture cross-lingual and code-mixed embeddings more efficiently. Hence, we fine-tune the MuRIL pretrained model on the ILID train set for our task.

5 Experimental Details

The machine learning models have been implemented using the Scikit-learn (Pedregosa et al., 2011) framework. Similarly, the FastText library [5] is utilized to implement the language identifiers of the texts modeled as text classifiers (Joulin et al., 2016b,a). The MuRIL (Khanuja et al., 2021) models are fine-tuned using the Huggingface transformer (Wolf et al., 2019) framework on an NVIDIA H100 GPU with 94GB RAM. The batch size, the number of epochs, the learning rate, maximum sequence length, and weight decay are set to 32, 10, 0.00002, 256, and 0.01 respectively. The fine-tuned MuRIL model is available at https://huggingface.co/pruthwik/ilid-muril-model.

6 Evaluation Metrics

All models have been evaluated using the macro F1 scores. The macro F1 score averages the F-1 scores across all languages. In order to evaluate the score the F1-score of each language, the precision and recall scores are also computed at the macro level. Each ensemble employs a voting mechanism to determine the predicted labels. Two types of voting mechanism are used: soft and hard. Hard voting is based on majority voting among the classifiers present in the ensemble. This type of voting is performed when the SVM classifier is part of the ensemble because SVM is inherently a non-probabilistic classifier. Soft voting aggregates the probability estimates of individual classifiers in the ensemble and the class with argmax of the sums of the predicted probabilities is chosen as the label. Ensembles where SVM is not included are

[5] https://fasttext.cc/docs/en/supervised-tutorial.html

Model	KNN	DT	RF	SVM	NB	LogReg	SGD	ILID		Bh-Ab
								Dev	Test	
Voting-1	✓	✓	✓		✓	✓		**1.00**	0.96	**1.00**
Voting-2	✓	✓	✓			✓	✓	**1.00**	0.96	**1.00**
Voting-3		✓	✓		✓	✓	✓	**1.00**	0.96	**1.00**
Voting-4	✓	✓	✓	✓	✓			0.99	0.96	**1.00**
Voting-5			✓	✓	✓	✓	✓	0.99	**0.99**	**1.00**
MuRIL								0.96	0.96	0.9
FastText								0.96	0.96	0.9

Table 5: Comparison of Voting Classifiers, MuRIL, and FastText on ILID dataset and Bhasha-Abhijnaanam (Bh-Ab) dataset. Highest scores are marked in bold.

evaluated using soft voting.

7 Results and Discussion

The ensemble machine learning models perform better than the individual models. The performance of the ensembles is also superior to the FastText and fine-tuned MuRIL models. The best five performing ensembles along with the FastText and fine-tuned MuRIL models are presented in Table 5 that represent the macro F1 score for each model. Although the scores are encouraging, the performance drops in languages that share a script such as Devanagari for Hindi, Maithili, Marathi, Konkani, and Sanskrit, Arabic for Kashmiri, Urdu, and Sindhi. We compare our models with the state-of-the-art IndicLID model (Madhani et al., 2023a) on the Bhasha-Abhijnaanam (Madhani et al., 2023a) benchmark of 88K sentences where our model outperforms the IndicLID model (Madhani et al., 2023a). The F1-scores of the IndicLID model on the Bhasha-Abhijnaanam dataset is 0.98 as reported in the paper. We could not run their fine-tuned IndicBERT model while the IndicLID FTN model has a macro F1-score of 0.88. Our deep learning baselines perform better on the languages where data are scraped, while the performance dips for languages that are created from other external resources.

7.1 Results for Individual Languages

The language-wise performance of the models is shown in Tables 6. We can observe from the table that the performance drops for extremely low resource languages such as Bodo, Dogri, Maithili, and Sindhi. Deep learning models suffer in identifying those languages for which they have scarcity during pre-training. MuRIL (Khanuja et al., 2021) outperforms every other model in languages on which the model was pre-trained.

8 Conclusion

In this paper, we present the ILID benchmark, a manually curated dataset for 23 languages, which includes English and 22 Indian languages containing 250K sentences. It is specifically designed for Indian language identification. Along with the dataset, we develop several machine learning an deep learning models that perform well, even with limited training data. This makes them especially suitable for low-resource languages. To check the consistency and reliability of our ML models, we compare them with deep learning models like MuRIL and FastText. The ILID dataset and models together provide important resources for advancing multilingual NLP research in Indian languages. We hope this work encourages more exploration and development in this area, which has been overlooked.

Lang	LR		DT		RF		SVM		SGD		KNN		NB		Muril		Fasttext	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
asm	0.98	**0.99**	0.96	0.96	0.98	0.98	0.98	**0.99**	0.97	0.98	0.90	0.96	0.97	0.98	**0.99**	**0.99**	0.96	0.96
ben	**1.00**	**1.00**	0.97	0.98	0.99	0.99	**1.00**	**1.00**	0.99	0.99	0.91	0.97	0.99	0.99	**1.00**	**1.00**	0.98	0.98
brx	**0.96**	**0.95**	0.91	0.89	**0.96**	0.94	**0.96**	**0.95**	0.94	**0.95**	0.85	0.93	**0.96**	**0.95**	0.91	0.91	0.92	0.91
doi	0.92	**0.92**	0.83	0.82	0.90	0.91	0.91	**0.92**	0.88	0.88	0.68	0.81	0.90	0.91	**0.93**	**0.92**	0.91	0.91
eng	**1.00**	**1.00**	0.99	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99	**1.00**	**1.00**	0.99	**1.00**	**1.00**	0.99	0.99
gom	**0.97**	**0.97**	0.84	0.83	0.94	0.95	0.96	**0.97**	0.90	0.91	0.78	0.90	0.94	0.96	**0.97**	**0.97**	0.91	0.92
guj	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.95	**1.00**	**1.00**	**1.00**	0.99	0.99
hin	0.99	0.98	0.93	0.92	0.98	0.97	0.99	**0.99**	0.97	0.97	0.80	0.92	**1.00**	0.95	0.99	**0.99**	0.98	0.98
kan	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.98	**1.00**	**1.00**	**1.00**	0.98	0.97
kas	**0.99**	0.98	0.94	0.94	0.97	0.98	**0.99**	0.99	0.94	0.95	0.96	0.96	0.85	0.98	**0.99**	**1.00**	0.97	0.96
mai	0.90	0.86	0.83	0.78	0.84	0.84	**1.00**	0.85	0.83	0.82	0.65	0.73	0.84	0.85	0.89	**0.88**	0.87	0.86
mal	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	0.86	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.95	**1.00**	**1.00**	**1.00**	0.94	0.94
mar	0.99	0.97	**1.00**	0.84	0.96	0.96	**1.00**	**0.98**	0.91	0.91	0.76	0.91	**1.00**	0.95	0.98	**0.98**	0.94	0.94
mni_Beng	**1.00**	**1.00**	0.99	0.99	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99
mni_Mtei	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.91	**1.00**	0.69	0.71	0.99	0.98
npi	0.92	0.91	0.85	0.84	0.91	0.90	1.00	0.91	0.87	0.86	0.75	0.83	1.00	0.91	**0.93**	**0.92**	0.88	0.87
ory	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.92	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
pan	**1.00**	**1.00**	0.99	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99	**1.00**	0.97	**1.00**	**1.00**	**1.00**	0.96	0.97
san	**1.00**	**1.00**	0.95	0.95	**1.00**	**1.00**	**1.00**	**1.00**	0.96	0.97	0.60	0.96	**1.00**	0.97	**1.00**	**1.00**	0.98	0.98
sat	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.99	0.92	**1.00**	0.70	0.72	0.99	0.99
snd_arab	**1.00**	**1.00**	0.97	0.98	0.99	**1.00**	**1.00**	**1.00**	0.99	**1.00**	0.99	0.97	0.99	0.98	**1.00**	**1.00**	**1.00**	0.99
snd_deva	0.94	0.93	0.88	0.85	0.92	0.91	0.94	0.92	0.89	0.90	0.80	0.88	0.92	0.92	0.94	0.94	0.92	0.92
tam	**1.00**	**1.00**	0.99	0.99	**1.00**	**1.00**	**1.00**	**1.00**	0.99	0.99	0.90	**1.00**	0.99	**1.00**	**1.00**	**1.00**	0.93	0.94
tel	**1.00**	**1.00**	0.99	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	0.97	0.97
urd	**1.00**	0.98	0.93	0.93	0.97	0.97	1.00	0.99	0.96	0.96	0.97	0.97	0.98	0.98	**1.00**	**1.00**	0.81	0.79

Table 6: F1-Scores on ILID Dev and Test Sets for each classifier on each language. Highest scores in each language in respective datasets across models are marked in bold

References

Milind Agarwal, Md Mahfuz Ibn Alam, and Antonios Anastasopoulos. 2023. LIMIT: Language identification, misidentification, and translation using hierarchical models in 350+ languages. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 14496–14519, Singapore. Association for Computational Linguistics.

Utsab Barman, Amitava Das, Joachim Wagner, and Jennifer Foster. 2014. Code mixing: A challenge for language identification in the language of social media. In *Proceedings of the first workshop on computational approaches to code switching*, pages 13–23.

Irshad Ahmad Bhat, Vandan Mujadia, Aniruddha Tammewar, Riyaz Ahmad Bhat, and Manish Shrivastava. 2014. Iiit-h system submission for fire2014 shared task on transliterated search. In *Proceedings of the 6th Annual Meeting of the Forum for Information Retrieval Evaluation*, pages 48–53.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the association for computational linguistics*, 5:135–146.

Alessio Bosca and Luca Dini. 2010. Language identification strategies for cross language information retrieval. In *CLEF (Notebook Papers/LABs/Workshops)*.

Isaac Caswell, Theresa Breiner, Daan van Esch, and Ankur Bapna. 2020. Language ID in the wild: Unexpected challenges on the path to a thousand-language web text corpus. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6588–6608, Barcelona, Spain (Online). International Committee on Computational Linguistics.

William B. Cavnar and John M. Trenkle. 1994. N-gram-based text categorization. *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online. Association for Computational Linguistics.

Alexis Conneau, Holger Schwenk, Loïc Barrault, and Yann Lecun. 2017. Very deep convolutional networks for text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1107–1116.

Najim Dehak, Pedro A Torres-Carrasquillo, Douglas A Reynolds, and Reda Dehak. 2011. Language recognition via i-vectors and dimensionality reduction. In *Interspeech*, pages 857–860.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Ted Dunning. 1994. Statistical identification of language. *Computing Science and Statistics*, 26:371–374.

David Elworthy. 1998. Language identification with confidence limits. In *Sixth Workshop on Very Large Corpora*.

Björn Gambäck and Amitav Das. 2017. Code-mixing in indian social media: A challenge for language identification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1003–1008.

Javier Gonzalez-Dominguez, Ignacio Lopez-Moreno, Hasim Sak, Joaquin Gonzalez-Rodriguez, and Pedro J Moreno. 2014. Automatic language identification using long short-term memory recurrent neural networks. In *Interspeech*, pages 2155–2159.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Armand Joulin, Edouard Grave, Piotr Bojanowski, Matthijs Douze, Hérve Jégou, and Tomas Mikolov. 2016a. Fasttext.zip: Compressing text classification models.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016b. Bag of tricks for efficient text classification.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, et al. 2021. Muril: Multilingual representations for indian languages. *arXiv preprint arXiv:2103.10730*.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Saurabh Kumar, Ranbir Sanasam, and Sukumar Nandi. 2023. IndiSocialFT: Multilingual word representation for Indian languages in code-mixed environment. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 3866–3871, Singapore. Association for Computational Linguistics.

Marco Lui and Timothy Baldwin. 2011. Cross-domain feature selection for language identification. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 553–561, Chiang Mai, Thailand. Asian Federation of Natural Language Processing.

Yash Madhani, Mitesh M. Khapra, and Anoop Kunchukuttan. 2023a. Bhasa-Abhijnaanam: Native-script and romanized language identification for 22 Indic languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 816–826, Toronto, Canada. Association for Computational Linguistics.

Yash Madhani, Sushane Parthan, Priyanka Bedekar, Gokul Nc, Ruchi Khapra, Anoop Kunchukuttan, Pratyush Kumar, and Mitesh Khapra. 2023b. 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.

Vandan Mujadia and Dipti Misra Sharma. 2024. Bhashaverse : Translation ecosystem for indian subcontinent languages.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in python. *the Journal of machine Learning research*, 12:2825–2830.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Xinying Song, Alex Salcianu, Yang Song, Dave Dopson, and Denny Zhou. 2021. Fast WordPiece tokenization. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2089–2103, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Karen Sparck Jones. 1972. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation.

textinspector. 2025. Statistics and readability scores - text inspector. https://textinspector.com/help/statistics-readability/. Accessed: 2025-07-27.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Liu Zhuang, Lin Wayne, Shi Ya, and Zhao Jun. 2021. A robustly optimized BERT pre-training approach with post-training. In *Proceedings of the 20th Chinese National Conference on Computational Linguistics*, pages 1218–1227, Huhhot, China. Chinese Information Processing Society of China.