CUFE@NLU of Devanagari Script Languages 2025: Language Identification using fastText

Michael Ibrahim
Computer Engineering Department, Cairo University
1 Gamaa Street, 12613
Giza, Egypt
michael.nawar@eng.cu.edu.eg

Abstract

Language identification is a critical area of research within natural language processing (NLP), particularly in multilingual contexts where accurate language detection can enhance the performance of various applications, such as machine translation, content moderation, and user interaction systems. This paper presents a language identification system developed using fastText. In the CHIPSAL@COLING 2025 Task on Devanagari Script Language Identification, the proposed method achieved first place, with an F_1 score of 0.9997.

1 Introduction

Language identification is crucial in Natural Language Processing (NLP), facilitating various applications like machine translation, information retrieval, and content filtering. Identifying languages with unique scripts, like Korean or Japanese, is fairly easy, but determining languages that use common scripts poses notable difficulties. An example of this is the Devanagari script, utilized by multiple languages, such as Hindi, Sanskrit, Marathi, Nepali, Bhojpuri, and more. Although these languages use the same script, they demonstrate significant differences in grammar, vocabulary, and morphology, resulting in a complex language identification challenge.

Conventional methods for language identification depend significantly on lexical characteristics and statistical models, usually needing extensive, domain-specific datasets to achieve good performance. Methods like n-gram modeling (Cavnar et al., 1994) or character-level classification (Zhang et al., 2015) have demonstrated effectiveness for certain languages but frequently underperform when utilized on intricate scripts such as Devanagari. This can be attributed in part to the morphological richness of languages that use Devanagari, where words may be significantly inflected, making it more challenging to differentiate languages based only on surface-level characteristics.

Recent advancements in neural network-based models have demonstrated substantial improvements in language identification tasks, especially when used on languages that share similar scripts. One such model is fastText (Joulin et al., 2017) that has attracted interest due to its capacity to model subword information, effectively capturing detailed morphological patterns and delivering strong performance even with smaller datasets or noisy text. These characteristics make fastText a compelling option for language identification tasks in scripts such as Devanagari, where languages have a considerable amount of lexical overlap yet differ in their subword configurations.

This paper describes a system that uses fastText for identifying languages in the Devanagari script. The developed system efficiently differentiates between languages written in Devanagari, despite their common orthographic traits, by utilizing fastText's capability to create word representations that encapsulate character-level n-grams. This system was trained and evaluated using the datasets provided by the CHIPSAL@COLING 2025 (Sarveswaran et al., 2025) Task on Devanagari Script Language Identification (Thapa et al., 2025).

The rest of the paper is organized as follows. The related work is summarized in Section 2. The dataset used for training and validation was detailed in Section 3. In Section 4, the system is presented. Section 5 summarizes the study's key findings.

2 Related Work

Several studies have explored language identification using various techniques. Traditional methods often rely on statistical models that analyze textual

Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL 2025), pages 273–277
January 19, 2025.

features such as n-grams or character frequencies. For instance, langid (Lui and Baldwin, 2012) can detect 97 languages and relies on a robust set of predefined features, which are calculated using Information Gain applied to different sets of n-grams. These features are used by a Naive Bayes classifier trained on a diverse corpus of text data from various sources. However, these traditional approaches may struggle with scripts like Devanagari due to shared vocabulary among languages.

FastText (Joulin et al., 2017) is one of the most popular models used for language identification. FastText uses a simpler linear classifier with a low-rank matrix constraint (Joulin et al., 2016). Its architecture incorporates hierarchical softmax, which helps reduce running time. Additionally, FastText combines a bag-of-words model with an N-gram approach to enhance performance and minimize processing time. While the N-gram model captures contextual character information around each instance but requires more memory, the bag-of-words model offers less detailed feature capture. By combining these two techniques, FastText creates a "bag-of-n-grams" model that balances performance and efficiency (Bojanowski et al., 2017).

CLD3 (Alex Salcianu, 2018) processes the input text by first extracting a range of n-grams, which are then transformed into dense vectors through an embedding layer. Each unique n-gram is represented by a fixed vector, and these vectors are averaged, with the frequency of each n-gram in the original text serving as the weighting factor. The resulting averaged vectors are concatenated and fed into a multi-layer perceptron, which generates a probability distribution over 107 possible languages.

One of the most powerful approaches today for language identification involves deep learning models like Long Short-Term Memory (LSTM) Networks (Schmidhuber et al., 1997). These models outperform older statistical and rule-based methods by effectively learning intricate patterns and capturing contextual relationships within the data. LSTM for Language Identification (Toftrup et al., 2021) applies Unicode-based written script identification, then for each script, a network is trained to predict the language based on the input text. In this approach, each character in the text is processed through an LSTM network, which then outputs a prediction for a single language. Finally, a max-pooling-based majority voting mechanism was used to combine the predictions from all characters and determine the dominant language of the input string.

Recently, hierarchical models were used for language identification. For instance, the LIMIT model (Agarwal et al., 2023) leverages layered structures to handle language identification, misidentification, and translation across over 350 languages. This method provides a comprehensive solution by integrating multiple layers of processing to enhance accuracy and robustness.

3 Dataset & Task

The shared task on Devanagari Script Language Identification (Thapa et al., 2025) aims to develop a system that can automatically determine the language of a sentence in Devanagari script among Nepali, Marathi, Sanskrit, Bhojpuri, and Hindi. This task addresses the critical need for accurate language identification in multilingual contexts. The data provided by this share task was sampled from different data sources:

- The Nepali data source came from 2 sources that focus on the Nepali election, (i) Nehate: Large-scale annotated data shedding light on hate speech in nepali local election discourse (Thapa et al., 2023) and (ii) Multi-Aspect Annotation and Analysis of Nepali Tweets on Anti-Establishment Election Discourse (Rauniyar et al., 2023).

- The Marathi data source, L3CubeMahaSent (Kulkarni et al., 2021), consists of almost 16,000 distinct tweets extracted from various Maharashtrian personalities' Twitter accounts, and it was annotated from sentiment analysis.

- The Sanskrit data source, Itihasa: A large-scale corpus for Sanskrit to English translation(Aralikatte et al., 2021), consists of 93,000 pairs of Sanskrit shlokas and their English translations.

- The Bhojpuri data source, English-Bhojpuri SMT System: Insights from the Karaka Model (Ojha, 2019), consists of 65,000 parallel sentences have been created containing 4,40,609 and 4,58,484 words in English and Bhojpuri respectively.

- The Hindi data source came from 2 sources that focuses on political hate speech in Indian election: (i) CHUNAV: Analyzing Hindi Hate

Language	Train	Development
Nepali	12544	2688
Marathi	11034	2364
Sanskrit	10996	2356
Bhojpuri	10184	2182
Hindi	7664	1643

Table 1: CHIPSAL@COLING 2025 Devanagari Script Language Identification Task - data split statistics.

Speech and Targeted Groups in Indian Election Discourse (Jafri et al., 2024) (ii) Uncovering political hate speech during Indian election campaign: A new low-resource dataset and baselines (Jafri et al., 2023)

The training dataset consists of a total of 52422 sentences distributed among Devanagari script languages as described in table 1. The development dataset consists of a total of 11233 sentences distributed among Devanagari script languages as described in table 1. Finally, the test dataset used for the evaluation of the developed method consists of a total of 11234 sentences.

4 Methodology, Results & Discussion

The training process will involve using fastText to create language models based on the training dataset. The following steps will be undertaken:

1. **Tokenization:** Tokenization is a fundamental step in the preprocessing phase of language identification, as it transforms raw text into manageable pieces. This process involves breaking down text into tokens, which can be words, phrases, or symbols, allowing for a more effective analysis of the sentence. In this work, the no language left behind [1] tokenizer (Costa-jussà et al., 2022) was used for tokenizing the text.

2. **Parameters Fine-Tuning:** The fastText classifier has 3 main parameters, (i) the words n-grams, (ii) the learning rate, and (iii) the number of training epochs. Different values for each of those parameters were examined, for the number of words n-gram values from 2 to 4 were examined, for the learning rate 2 values of 0.05 and 0.1 were examined, and for the number of training epochs 2 values of 25 and 50 were examined. For each combination of these parameters, a fastText classifier was trained on the 52422 sentences of the training set, and its performance was evaluated on the 11233 sentences of the development set. The F_1 scores of the developed models are summarized in table 2.

Ngrams	lr	Epochs	F_1
2	0.05	25	0.9968
2	0.05	50	0.9981
2	0.1	25	0.9975
2	0.1	50	0.9981
3	0.05	25	0.9961
3	0.05	50	0.9971
3	0.1	25	0.9971
3	0.1	50	0.9978
4	0.05	25	0.9952
4	0.05	50	0.9965
4	0.1	25	0.9963
4	0.1	50	0.9973

Table 2: Results of the development set.

3. **Final Model Training:** After analyzing the results of the experiments summarized in table 2, the final model was trained on 63655 sentences that represent the entire training and development sets, the fastText classifier used 2-gram word feature, a learning rate of 0.1, and was trained for 50 epochs. This model was trained on a Google colab CPU machine with a total memory of 12.67 GB, the training of this model took less than 90 seconds, and the generation of the label to the 11234 sentences of the test set took less than 10 seconds.

The developed system has 2 main limitations:

1. Many Devanagari languages were not considered in this shared task (Garhwali, Kashmiri, etc.), and when the number of languages to be identified by a classifier increases, the average accuracy generally tends to decrease (Leong et al., 2022). The effect of the change in the number of languages on the performance of the fastText classifier was not assessed in this study.

2. The developed fastText classifier has a very good performance since the data used for training and evaluating the model came from similar data sources, however, the fastText model

[1] https://huggingface.co/facebook/nllb-200-distilled-600M

uses simple features like word n-grams, so, it might learn to classify a sentence into a given language based on something like a proper noun. So, if there was a Nepali tweet discussing the Indian election might be wrongfully classified as as Hindi. The effect of out-of-sample was also not assessed in this study.

5 Conclusions

In this paper, a fastText classifier was trained to identify the Devanagari script language from 5 different Languages: Nepali, Marathi, Sanskrit, Bhojpuri, and Hindi. The proposed method is efficient as the final model training and the generation of the labels for the test set takes less than 2 minutes on a CPU machine, and it was ranked first on the CHIPSAL@COLING 2025 Task on Devanagari Script Language Identification achieving an F_1 score of 0.9997.

References

Milind Agarwal, Md Mahfuz Ibn Alam, and Antonios Anastasopoulos. 2023. Limit: Language identification, misidentification, and translation using hierarchical models in 350+ languages. *arXiv preprint arXiv:2305.14263*.

Anton Bakalov Chris Alberti Daniel Andor David Weiss Emily Pitler Greg Coppola Jason Riesa Kuzman Ganchev et al. Alex Salcianu, Andy Golding. 2018. . compact language detector v3.

Rahul Aralikatte, Miryam De Lhoneux, Anoop Kunchukuttan, and Anders Søgaard. 2021. Itihasa: A large-scale corpus for sanskrit to english translation. In *Proceedings of the 8th Workshop on Asian Translation (WAT2021)*, pages 191–197.

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.

William B Cavnar, John M Trenkle, et al. 1994. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd annual symposium on document analysis and information retrieval*, volume 161175, page 14. Ann Arbor, Michigan.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

Farhan Ahmad Jafri, Kritesh Rauniyar, Surendrabikram Thapa, Mohammad Aman Siddiqui, Matloob Khushi, and Usman Naseem. 2024. Chunav: Analyzing hindi hate speech and targeted groups in indian election discourse. *ACM Transactions on Asian and Low-Resource Language Information Processing*.

Farhan Ahmad Jafri, Mohammad Aman Siddiqui, Surendrabikram Thapa, Kritesh Rauniyar, Usman Naseem, and Imran Razzak. 2023. Uncovering political hate speech during indian election campaign: A new low-resource dataset and baselines.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, pages 427–431. Association for Computational Linguistics.

Atharva Kulkarni, Meet Mandhane, Manali Likhitkar, Gayatri Kshirsagar, and Raviraj Joshi. 2021. L3cubemahasent: A marathi tweet-based sentiment analysis dataset. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 213–220.

Colin Leong, Joshua Nemecek, Jacob Mansdorfer, Anna Filighera, Abraham Owodunni, and Daniel Whitenack. 2022. Bloom library: Multimodal datasets in 300+ languages for a variety of downstream tasks. *arXiv preprint arXiv:2210.14712*.

Marco Lui and Timothy Baldwin. 2012. langid. py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 system demonstrations*, pages 25–30.

Atul Kr Ojha. 2019. English-bhojpuri smt system: Insights from the karaka model. *arXiv preprint arXiv:1905.02239*.

Kritesh Rauniyar, Sweta Poudel, Shuvam Shiwakoti, Surendrabikram Thapa, Junaid Rashid, Jungeun Kim, Muhammad Imran, and Usman Naseem. 2023. Multi-aspect annotation and analysis of nepali tweets on anti-establishment election discourse. *IEEE Access*.

Kengatharaiyer Sarveswaran, Bal Krishna Bal, Surendrabikram Thapa, Ashwini Vaidya, and Sana Shams. 2025. A brief overview of the first workshop on challenges in processing south asian languages (chipsal). In *Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL)*.

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

Surendrabikram Thapa, Kritesh Rauniyar, Farhan Ahmad Jafri, Surabhi Adhikari, Kengatharaiyer Sarveswaran, Bal Krishna Bal, Hariram Veeramani,

and Usman Naseem. 2025. Natural language understanding of devanagari script languages: Language identification, hate speech and its target detection. In *Proceedings of the First Workshop on Challenges in Processing South Asian Languages (CHiPSAL)*.

Surendrabikram Thapa, Kritesh Rauniyar, Shuvam Shiwakoti, Sweta Poudel, Usman Naseem, and Mehwish Nasim. 2023. Nehate: Large-scale annotated data shedding light on hate speech in nepali local election discourse. In *ECAI 2023*, pages 2346–2353. IOS Press.

Mads Toftrup, Søren Asger Sørensen, Manuel R. Ciosici, and Ira Assent. 2021. A reproduction of apple's bi-directional LSTM models for language identification in short strings. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Student Research Workshop*, pages 36–42, Online. Association for Computational Linguistics.

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. *Advances in neural information processing systems*, 28.