# 1-800-SHARED-TASKS @ NLU of Devanagari Script Languages: Detection of Language, Hate Speech, and Targets using LLMs

**Jebish Purbey**
Pulchowk Campus, IoE
jebishpurbey@gmail.com

**Siddartha Pullakhandam**
University of Wisconsin
pullakh2@uwm.edu

**Kanwal Mehreen \***
Traversaal.ai
kanwal@traversaal.ai

**Muhammad Arham \***
NUST/Traversaal.ai
arhamm40182@gmail.com

**Drishti Sharma \***
Cohere For AI Community
drishtishrma@gmail.com

**Ashay Srivastava**
University of Maryland
ashays06@umd.edu

**Ram Mohan Rao Kadiyala**
University of Maryland, College Park
rkadiyal@umd.edu

## Abstract

This paper presents a detailed system description of our entry for the CHiPSAL 2025 shared task, focusing on language detection, hate speech identification, and target detection in Devanagari script languages. We experimented with a combination of large language models and their ensembles, including MuRIL, IndicBERT, and Gemma-2, and leveraged unique techniques like focal loss to address challenges in the natural understanding of Devanagari languages, such as multilingual processing and class imbalance. Our approach achieved competitive results across all tasks: F1 of 0.9980, 0.7652, and 0.6804 for Sub-tasks A, B, and C respectively. This work provides insights into the effectiveness of transformer models in tasks with domain-specific and linguistic challenges, as well as areas for potential improvement in future iterations.

## 1 Introduction

Large language models (LLMs) have revolutionized natural language processing (NLP) yet South Asian languages remain largely underrepresented within these advancements despite being home to over 700 languages, 25 major scripts, and approximately 1.97 billion people. Addressing these gaps, this paper focuses on three critical NLP tasks of CHiPSAL 2025 (Sarveswaran et al., 2025) in

Devanagari-scripted languages: 5-way classification of the text based on the language of the text (Sub-task A), Binary classification for detecting hate speech in the text (Sub-task B), and 3-way classification for detecting target of hate speech in a text (Sub-task C) (Thapa et al., 2025). Our system leverages the multilingual capabilities of open-source LLMs namely IndicBERT V2 (Doddapaneni et al., 2023), MuRIL (Khanuja et al., 2021), and Gemma-2 (GemmaTeam, 2024) and their ensembles for natural language understanding of Devanagari script languages. Our work contributes to advancing language technology in South Asia, aiming for inclusivity and deeper understanding across diverse linguistic landscapes.

## 2 Dataset & Task

The goal of Sub-task A is to determine the language of the given Devanagari script among the 5 languages to address the critical need for accurate multilingual identification. The dataset consists of text in Nepali (Thapa et al., 2023; Rauniyar et al., 2023), Marathi (Kulkarni et al., 2021), Sanskrit (Aralikatte et al., 2021), Bhojpuri (Ojha, 2019), and Hindi (Jafri et al., 2024, 2023). For Sub-task B, the goal is to determine if the text contains hate speech or not. The dataset consists of social media text (tweets) in Hindi and Nepali languages. Sub-task C follows Sub-task B, where the goal is to identify the targets of hate speech among "individual", "or-

---

* equal contribution

ganization", or "community". Similar to Sub-task B, the dataset for Sub-task C is in Hindi and Nepali languages. The distribution of labels for the three datasets can be seen in table 1, 2, and 3 respectively.

| Class | Train | Dev | Test |
|---|---|---|---|
| Nepali | 12544 | 2688 | 2688 |
| Marathi | 11034 | 2364 | 2365 |
| Sanskrit | 10996 | 2356 | 2356 |
| Bhojpuri | 10184 | 2182 | 2183 |
| Hindi | 7664 | 1643 | 1642 |
| Total | 52422 | 11233 | 11234 |

Table 1: Class distribution for Subtask A

| Class | Train | Dev | Test |
|---|---|---|---|
| Non-hate | 16805 | 3602 | 3601 |
| Hate | 2214 | 474 | 475 |
| Total | 19019 | 4076 | 4076 |

Table 2: Class distribution for the Subtask B

| Class | Train | Dev | Test |
|---|---|---|---|
| Individual | 1074 | 230 | 230 |
| Organization | 856 | 183 | 184 |
| Community | 284 | 61 | 61 |
| Total | 2214 | 474 | 475 |

Table 3: Class distribution for the Subtask C

## 3 Methodology

The common approach to all three Sub-tasks was to fine-tune a multitude of multilingual models in the train set and use the dev set to select the best few models during the Evaluation phase. The selected best models were then fine-tuned again on both the train and dev sets and their ensemble, by majority voting, was used for the final prediction of the test set during the Testing phase as shown in Figure 1. The models fine-tuned under this approach include decoder-only models such as Gemma-2 9B, Llama 3.1 8B (LlamaTeam, 2024), and Mistral Nemo Base 12B (MistralAI, 2024), and BERT (Devlin et al., 2019) based models such as IndicBERT V2, MuRIL, XLM Roberta (Conneau et al., 2019), mDistilBERT (Sanh et al., 2019) and mBERT (Devlin et al., 2018). For decoder-only models, each Sub-task was formulated as a text-generation task where each model was asked to generate only one option among the given choices. For BERT-based models, each Sub-task was formulated as a multi-label classification task by adding a classification head to the model.

For Sub-task A, each decoder-only models were trained for 1 epoch with a learning rate of 2e-4. The BERT-based models were trained for 5 epochs with a learning rate of 4e-5 with weighted cross-entropy loss. For Sub-task B, decoder-only models were trained for 2-4 epochs with a learning rate of 2e-4. The BERT-based models were trained for 5 epochs with a learning rate of 4e-5.

To handle the class imbalance in sub-task B, focal loss (Lin et al., 2018) was used for BERT-based models. Focal loss modifies cross-entropy by reducing the relative loss for well-classified exam-

ples, focusing more on hard, misclassified examples. The focal loss is given by formula 1:

$$\mathcal{L}_{\text{focal}} = -\alpha_t(1 - p_t)^\gamma \log(p_t) \qquad (1)$$

Where, $\alpha_t$ is the balancing factor for class $t$, $p_t$ is the model's estimated probability for the correct class, and $\gamma$ is the focusing parameter that adjusts the rate at which easy examples are down-weighted. The hyperparameters $\alpha_t$ and $\gamma$ were determined using grid search as 0.35 and 4.0 respectively.

For Sub-task C, only decoder models were used during the Testing phase as BERT-based models massively underperformed in limited tests. An additional Gemma-2 27B model was fine-tuned for Sub-task B and C using Odds Ratio Preference Optimization (ORPO) (Hong et al., 2024) for better alignment. All the fine-tuning of decoder-only models was carried out using Unsloth with Low-Rank Adaptation of Large Language Models (LoRA) (Hu et al., 2021). The rank ($r$) and alpha ($\alpha$) values used were 16 for both.

| Model | F1 | Recall | Precision |
|---|---|---|---|
| mBERT | 0.9962 | 0.9962 | 0.9962 |
| mDistilBERT | 0.9955 | 0.9957 | 0.9954 |
| XLM Roberta | 0.9965 | 0.9966 | 0.9964 |
| MuRIL | **0.9978** | **0.9978** | **0.9977** |
| IndicBERT V2 | **0.9978** | **0.9978** | **0.9977** |
| Llama 3.1 8B | 0.9957 | 0.9957 | 0.9958 |
| Gemma-2 9B | 0.9965 | 0.9965 | 0.9965 |
| Mistral Nemo 12B | 0.9962 | 0.9962 | 0.9961 |

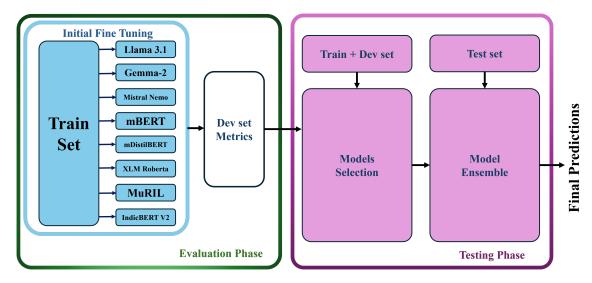Table 4: Performance metrics for Subtask A on the dev set

Figure 1: System design workflow. The development set is initially used to select the best-performing models, which are then retrained on the combined train and development set. Selected models are ensembled to generate final predictions on the test set.

| Model | Description | F1 |
|---|---|---|
| MuRIL | Fine-tuned on train+dev set | 0.9968 |
| IndicBERT V2 | Fine-tuned on train+dev set | 0.9977 |
| Gemma-2 9B | Fine-tuned on train+dev set | 0.9973 |
| Ensemble-1 | MuRIL's prediction as fallback in case of no majority | 0.9979 |
| Ensemble-2 | IndicBERT V2's prediction as fallback in case of no majority | **0.9980** |
| Ensemble-3 | Gemma-2 9B's prediction as fallback in case of no majority | 0.9979 |

Table 5: Performance metrics for Subtask A on the test set

| Model | F1 | Recall | Precision |
|---|---|---|---|
| mBERT | 0.7142 | 0.7152 | 0.7133 |
| mDistilBERT | 0.6286 | 0.6093 | 0.6668 |
| XLM Roberta | 0.7182 | 0.7367 | 0.7037 |
| MuRIL | 0.6773 | **0.7741** | 0.6530 |
| IndicBERT V2 | 0.7298 | 0.7215 | 0.7392 |
| Gemma-2 9B | 0.7094 | 0.6677 | **0.8051** |
| Gemma-2 9B (Few-shot) | **0.7412** | 0.7019 | 0.7929 |

Table 6: Performance metrics for Subtask B on dev set

| Model | F1 | Recall | Precision |
|---|---|---|---|
| IndicBERT V2 | 0.7582 | 0.7732 | 0.7455 |
| Gemma-2 9B (Few-shot) | 0.7588 | 0.7360 | 0.7895 |
| Gemma-2 27B Orpo | 0.7494 | 0.7261 | 0.7814 |
| Ensemble | **0.7652** | 0.7441 | 0.7925 |

Table 7: Performance metrics for Subtask B on test set

| Model | F1 | Recall | Precision |
|---|---|---|---|
| mDistilBERT | 0.4173 | 0.4296 | 0.4560 |
| mBERT | 0.4398 | 0.4567 | 0.4926 |
| XLM Roberta | 0.5455 | 0.5765 | 0.5528 |
| IndicBERT V2 | 0.4639 | 0.4648 | 0.4643 |
| Gemma-2 9B | **0.6937** | **0.6691** | **0.7520** |

Table 8: Performance metrics for Subtask C on dev set

## 4 Results and Discussion

### 4.1 Evaluation Phase

During the Evaluation phase, various models were assessed across Sub-tasks A, B, and C using the dev set to identify the top-performing models for each task. For Sub-task A (Table 4), the BERT-based models and decoder-only models, both delivered strong performances, with IndicBERT V2 and MuRIL emerging as the best models, each achieving an F1 score of 0.9978. They also had high recall and precision, indicating their robustness in effectively balancing sensitivity and specificity in task A classification. mBERT, XLM-Roberta, and larger generative models like Gemma-2 and Mistral Nemo also scored close to the top contenders, demonstrating that BERT-based and recent LLMs both possess considerable ability in text classification. For Sub-task B (Table 6), models' performance varied more significantly, reflecting the increased complexity compared to Sub-task A. Among the evaluated models, fine-tuned Gemma-2 9B with few-shot prompting yielded an F1 score of 0.7412. This shows Gemma-2's effective adap-

| Model | Description | F1 | Recall | Precision |
|---|---|---|---|---|
| Gemma-2 9B | Fine-tuned on train+dev set with learning rate 2e-4 and batch size of 4 for 2 epochs | 0.6213 | 0.6084 | 0.6734 |
| Gemma-2 9B | Fine-tuned on train+dev set with learning rate 2e-4 and batch size of 2 for 2 epochs | 0.6503 | 0.6371 | 0.6982 |
| Gemma-2 27B | Fine-tuned on train+dev set using ORPO with a batch size of 8 for 1 epoch | **0.6804** | **0.6669** | **0.7183** |

Table 9: Performance metrics for task C on the test set

tation in low-resource scenarios even with limited examples. IndicBERT V2 and XLM-Roberta also provided competitive results, with IndicBERT V2 achieving an F1 score of 0.7298, reinforcing its efficacy across both tasks. This marked Gemma-2 9B and IndicBERT V2 as the top choices to be further evaluated for Sub-task B during the Testing phase. In Sub-task C (Table 8), Gemma-2 9B demonstrated superior results with an F1 score of 0.6937. This outcome was significantly better than all other models, indicating Gemma-2's robust performance for tasks with limited examples. XLM Roberta achieved the second-highest F1 score of 0.5455. The performance of other models shows the complexity of the task as except for Gemma-2, other models couldn't cross the F1 score of 0.6.

## 4.2 Testing Phase

For the testing phase, we retrained the top-selected models from the Evaluation phase by incorporating both the train and dev sets to create a more generalized model for final testing. For Sub-task A (Table 5), ensemble techniques were applied to enhance accuracy further, leading to notable improvements in performance. Three ensembles were constructed, each with a different fallback model for cases without a majority prediction. Among these, Ensemble-2, which defaulted to IndicBERT V2's predictions when no majority was reached, yielded the highest F1 score of 0.9980. This ensemble strategy was instrumental in refining classification outcomes by leveraging the strengths of multiple models while relying on IndicBERT V2's consistency as a fallback. As a result, Sub-task A saw an optimal performance boost, indicating the success of ensembling techniques in improving classification tasks with high base accuracy. For Sub-task B (Table 7), we employed a similar ensemble approach to maximize prediction performance. Ensemble results demonstrated improved robustness and balance across the metrics, culmi-

nating in an F1 score of 0.7652, with strong recall (0.7441) and precision (0.7925). For the ensemble, we employed an additional Gemma-2 27B trained using ORPO with the two models selected during the Evaluation phase. The overall gains from the ensemble approach for this task underscore its potential to improve tasks with more nuanced, challenging data patterns. In Sub-task C (Table 9), instead of using ensembling, we selected Gemma-2 27B ORPO as the optimal model for its strong performance during testing. This model achieved an F1 score of 0.6804, with balanced recall (0.6669) and precision (0.7183), showcasing its capability to handle more granular classification without the need for ensemble interventions. The decision to forego ensembling was based on the observation that Gemma-2 27B's setup offered robust, reliable performance on its own, suggesting that, for some tasks, a single, finely-tuned model can sometimes match or exceed ensemble outcomes.

## 5 Conclusion

Our results demonstrate the importance of leveraging tailored approaches to tackle complex natural language understanding tasks across multiple languages in Devanagari script. By combining the multilingual strengths of the BERT-based models, focal loss for class sensitivity, and the generative power of Gemma-2, we achieved notable performance improvements across the subtasks. These findings highlight the value of adapting model architectures and training strategies to the nuances of each task, especially in handling multilingual contexts and imbalanced classes. This work lays a foundation for more refined, scalable hate speech detection systems for South Asian languages that can respond effectively to diverse and complex online discourse.

## Limitations

The datasets used for training and evaluation in hate speech and target detection are relatively small, which may impact the generalizability of the models in real-world applications. The challenges such as unbalanced datasets, difficulties in data collection, and issues with code-mixed languages, as noted in prior research (Parihar et al., 2021), remain significant hurdles in the accurate detection of hate speech. Although techniques like focal loss and Odds Ratio Preference Optimization (ORPO) were applied to improve performance, the models still struggle with fine-grained distinctions in ambiguous hate speech contexts. Additionally, the decoder-only models were trained in 4-bit precision due to computational limitations, and they may perform better in full-precision mode. While these models performed well in most tasks, they are computationally intensive, requiring substantial resources for both fine-tuning and inference. On the other hand, BERT-based models performed well in Sub-tasks A and B, and with larger datasets, they may offer better performance for Sub-task C at a lower computational cost than decoder-only models.

## Ethical Considerations

When developing models for detecting hate speech and its targets, it's important to address several ethical concerns. A major issue is the potential for bias in both the data and the model's outputs. Since the datasets used in the development are limited and might not fully represent all social contexts, there's a risk that the models could unintentionally reinforce biases or target specific groups unfairly. These models might also be used in ways that could cause harm, such as censoring or flagging content incorrectly without human oversight. Given the complex nuances of hate speech, it's crucial to avoid over-censorship, which may otherwise lead to the unjust targeting of certain communities or the stifling of legitimate free speech.

## References

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.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *CoRR*, abs/1911.02116.

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

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *Preprint*, arXiv:1810.04805.

Sumanth Doddapaneni, Rahul Aralikatte, Gowtham Ramesh, Shreya Goyal, Mitesh M. Khapra, Anoop Kunchukuttan, and Pratyush Kumar. 2023. Towards leaving no Indic language behind: Building monolingual corpora, benchmark and models for indic languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12402–12426, Toronto, Canada. Association for Computational Linguistics.

GemmaTeam. 2024. Gemma: Open models based on gemini research and technology. *Preprint*, arXiv:2403.08295.

Jiwoo Hong, Noah Lee, and James Thorne. 2024. Orpo: Monolithic preference optimization without reference model. *Preprint*, arXiv:2403.07691.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *Preprint*, arXiv:2106.09685.

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.

Simran Khanuja, Diksha Bansal, Sarvesh Mehtani, Savya Khosla, Atreyee Dey, Balaji Gopalan, Dilip Kumar Margam, Pooja Aggarwal, Rajiv Teja Nagipogu, Shachi Dave, Shruti Gupta, Subhash Chandra Bose Gali, Vish Subramanian, and Partha Talukdar. 2021. Muril: Multilingual representations for indian languages. *Preprint*, arXiv:2103.10730.

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.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. 2018. Focal loss for dense object detection. *Preprint*, arXiv:1708.02002.

LlamaTeam. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

MistralAI. 2024. Mistral nemo.

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

Anil Singh Parihar, Surendrabikram Thapa, and Sushruti Mishra. 2021. Hate speech detection using natural language processing: Applications and challenges. In *2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)*, pages 1302–1308. IEEE.

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*.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108.

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)*.

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.

# A Appendix

## A.1 Confusion Matrix

We provide the confusion matrix for all the models we tested below:



Figure 2: mBERT's Confusion Matrix for Language Detection

### A.1.1 Sub-task A: Language Detection

**Evaluation Phase**



Figure 3: mDistilBERT's Confusion Matrix for Language Detection



Figure 4: XLM Roberta's Confusion Matrix for Language Detection

Figure 5: MuRIL's Confusion Matrix for Language Detection



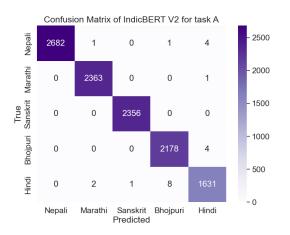Figure 8: Gemma-2 9B's Confusion Matrix for Language Detection



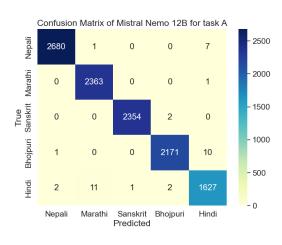Figure 6: IndicBERT V2's Confusion Matrix for Language Detection



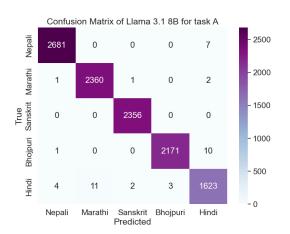Figure 9: Mistral Nemo's Confusion Matrix for Language Detection

**Testing Phase**



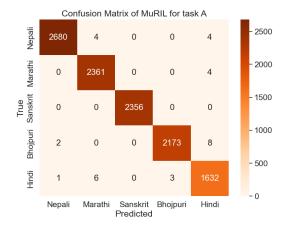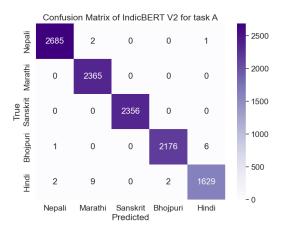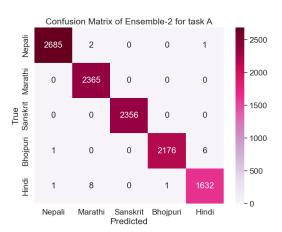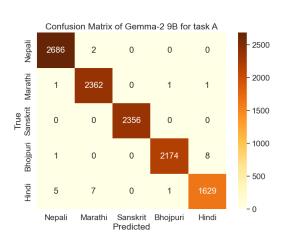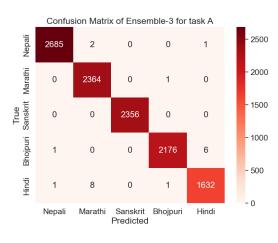Figure 7: Llama 3.1 8B's Confusion Matrix for Language Detection



Figure 10: MuRIL's Confusion Matrix for Language Detection

Figure 11: IndicBERT V2's Confusion Matrix for Language Detection



Figure 14: Ensemble-2's Confusion Matrix for Language Detection



Figure 12: Gemma-2 9B's Confusion Matrix for Language Detection



Figure 15: Ensemble-3's Confusion Matrix for Language Detection

### A.1.2 Sub-task B: Hate Speech Detection

**Evaluation Phase**



Figure 13: Ensemble-1's Confusion Matrix for Language Detection



Figure 16: mBERT's Confusion Matrix for Hate Speech Detection

Figure 17: mDistilBERT's Confusion Matrix for Hate Speech Detection



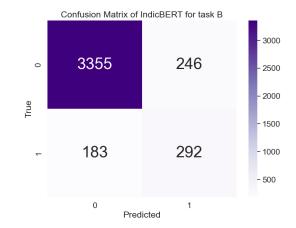Figure 20: Gemma-2 9B's Confusion Matrix for Hate Speech Detection



Figure 18: XLM Roberta's Confusion Matrix for Hate Speech Detection



Figure 21: Gemma-2 9B (Few-shot)'s Confusion Matrix for Hate Speech Detection

**Testing Phase**



Figure 19: IndicBERT V2's Confusion Matrix for Hate Speech Detection



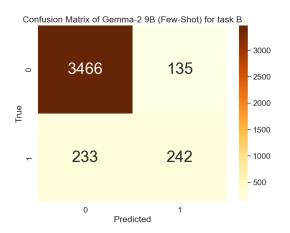Figure 22: IndicBERT V2's Confusion Matrix for Hate Speech Detection

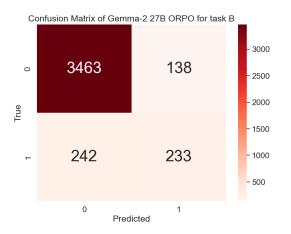Figure 23: Gemma-2 9B (Few-shot)'s Confusion Matrix for Hate Speech Detection



Figure 26: mBERT's Confusion Matrix for Hate Speech Target Detection



Figure 24: Gemma-2 27B ORPO's Confusion Matrix for Hate Speech Detection



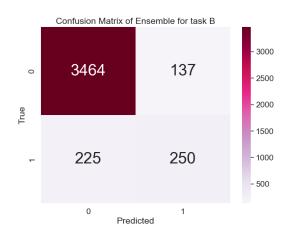Figure 27: mDistilBERT's Confusion Matrix for Hate Speech Target Detection



Figure 25: Ensemble's Confusion Matrix for Hate Speech Detection

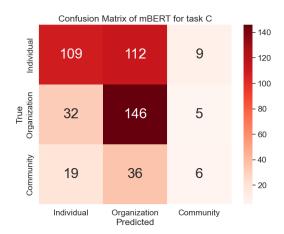### A.1.3 Sub-task C: Hate Speech Target Detection

**Evaluation Phase**



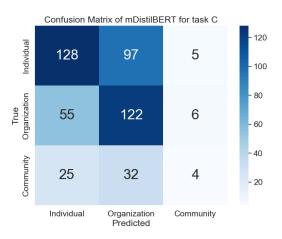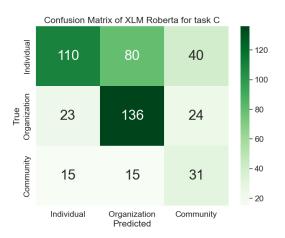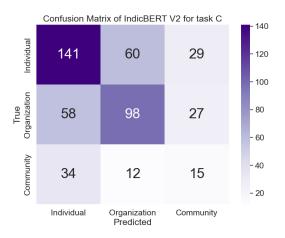Figure 28: XLM Roberta's Confusion Matrix for Hate Speech Target Detection

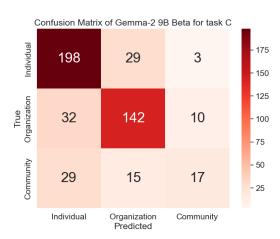Figure 29: IndicBERT V2's Confusion Matrix for Hate Speech Target Detection



Figure 30: Gemma-2 9B's Confusion Matrix for Hate Speech Target Detection

**Testing Phase**



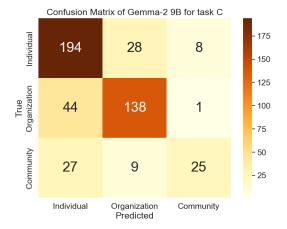Figure 31: Gemma-2 9B Alpha's Confusion Matrix for Hate Speech Target Detection



Figure 32: Gemma-2 9B Beta's Confusion Matrix for Hate Speech Target Detection



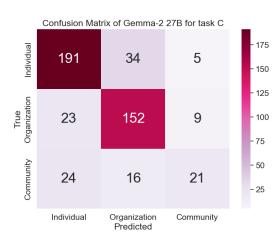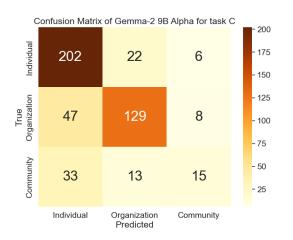Figure 33: Gemma-2 27B's Confusion Matrix for Hate Speech Target Detection

## A.2 System Replication

We provide the details of hyperparameters used in training for replicating the process in Table 10 and 11.

| Hyperparameter | Value |
|---|---|
| Learning rate | 2e-4 |
| Learning rate scheduler | linear |
| Weight decay | 0.01 |
| LoRA rank | 16 |
| LoRA alpha | 16 |
| LoRA dropout | 0 |
| **Language Detection** | |
| Max length (tokens) | 2048 |
| Batch size | 9 |
| Gradient accumulation | 3 |
| Warmup steps | 5 |
| Num of epochs | 1 |
| **Hate Speech Detection** | |
| Max length (tokens) | 1024 |
| Batch size | 16 |
| Gradient accumulation | 1 |
| Warmup steps | 10 |
| Num of epochs | 2-4 |
| **Hate Speech Target Detection** | |
| Max length (tokens) | 1024 |
| Batch size | 2-4 |
| Gradient accumulation | 1 |
| Warmup steps | 0 |
| Num of epochs | 2 |

Table 10: Hyperparameters' values for decoder-only models across tasks

| Hyperparameter | Values |
|---|---|
| **Max length of input sequence** | 64 |
| **Batch size** | 512 |
| **Num of workers** | 2 |
| **Num of epochs** | 5 |
| **Learning rate** | 4e-5 |
| **Learning rate scheduler** | linear |
| **Focal loss Alpha** | 0.35 |
| **Focal loss Gamma** | 4.0 |

Table 11: Hyperparameters' values for BERT-based models

Table 10 presents the hyperparameters for decoder-only models across tasks, with core values, such as learning rate, weight decay, and LoRA values shared across tasks. Task-specific parameters like maximum token length, batch size, gradient accumulation, warmup steps, and epochs were experimented with to meet the requirements of each task. For hyperparameters not listed, default values were used for each model.

## A.3 Prompts

The prompts used for decoder-only models are provided below:

### A.3.1 Task A: Language Detection

```
Task: You are an expert linguist specializing
    in Devanagari script languages. Your task
    is to identify the language of the given
    text.

### Instruction:
Analyze the following Devanagari script text
    and determine its language. Choose the
    correct language code from these options:
0: Nepali
1: Marathi
2: Sanskrit
3: Bhojpuri
4: Hindi

### Input:
Text: {text}

### Response:
The language code for the given text is: {label}
```

### A.3.2 Task B: Hate Speech Detection

```
Task: You are fluent in Nepali and Hindi
    languages. Your task is to classify if the
    given input text contains hate speech or
    not.

### Instruction:
The goal of this subtask is to identify the
    targets of hate speech in a given text.
    Choose the correct category from these
    options:
1: Hate
0: Non-Hate

### Examples:
Input: {example_text1}
Response: {example_text1_label}

Input: {example_text2}
Response: {example_text2_label}

Input: {example_text3}
Response: {example_text3_label}

Input: {example_text4}
Response: {example_text4_label}

Input: {example_text5}
Response: {example_text5_label}

### Input:
{text}

### Response:
{label}
```

### A.3.3 Task C: Hate Speech Target Detection

You are an expert linguist specializing in
    detecting hate speech targets in
    Devanagari-script tweets. Your task is to
    classify the target of hate speech.

### Instruction:
Analyze the given tweet in Devanagari script
    and determine who the hate speech is
    targeting.

Step 1: First, decide if the target is an
    individual or a group.
Step 2 (if group): If it's a group, further
    classify it as either an organization or a
    community.

Classify the final label according to these
    categories:
0. Individual: A specific person or a small set
    of identifiable individuals
1. Organization: A formal entity, institution,
    or company
2. Community: A broader group based on
    ethnicity, religion, gender, or other
    shared characteristics

### Input:
{}

### Response:
{}