VATIKA-QA: A Hybrid BERT-IndicBART Approach for Hindi Question Answering in Tourism Domain

Prudvi Kumar Reddy Narreddi Subbannagari[1,*], Avanish S Velidi[2] and Anand Kumar Madasamy[2]

[1]Mohan Babu University, Tirupati, Andhra Pradesh, India
[2]Department of Information Technology, National Institute of Technology, Surathkal, India

Abstract

This paper describes the development and evaluation of an innovative hybrid encoder-decoder model for Question Answering (QA) in Hindi, focusing on the tourism sector. The importance of this research is emphasized by the increasing demand for accessible information systems, especially for languages with limited digital resources like Hindi in culturally significant regions such as Varanasi. The proposed framework uses the l3cube-pune/hindi-bert-v2 model as an effective encoder to interpret Hindi text and the ai4bharat/IndicBART model as a decoder to generate answers in natural language. The core of this system is the VATIKA dataset, a comprehensive Hindi QA resource created for machine reading comprehension within the Varanasi tourism context, covering ten different domains. The methodology involves integrating BERT and IndicBART through a linear projection layer. An additional strategy explored was the use of Named Entity Recognition (NER) to explicitly tag entities. Experimental results demonstrate the high performance of the basic BERT-IndicBART hybrid model. However, an important finding was that adding NER-based tags to the input unexpectedly caused a consistent decrease in performance across all metrics, including Exact Match, F1 Score, BLEU, and ROUGE. This paper discusses the implications of this counterintuitive result, providing empirical data on the challenges of incorporating external knowledge into QA systems for low-resource languages.

Keywords

Question Answering (QA), Natural Language Processing (NLP), Hindi, Tourism, Encoder-Decoder Models, BERT, IndicBART, Multilingual QA, Information Retrieval, Varanasi, VATIKA Dataset

1. Introduction

As a key contributor to the Indian economy, tourism is instrumental in generating revenue, creating employment, and sustaining local enterprises. It also fosters cultural interchange, aids in the preservation of heritage, and stimulates the development of infrastructure. Varanasi, one of the world's oldest continually inhabited cities and a prominent spiritual and cultural hub, is a standout among India's numerous tourist sites. The city attracts millions of visitors annually for spiritual and cultural pursuits, presenting distinct information access challenges due to its specialized terminology, cultural richness, and location-specific logistics (e.g., Ganga Aarti, Kunds, Ashrams). A standard QA system would find it difficult to grasp these subtleties, which underscores the necessity for a solution that is both culturally aware and specific to the domain. Creating such a system for Varanasi could establish a model for other culturally rich and linguistically diverse areas.

This research originates from the "VATIKA: Varanasi Tourism in Question Answer System (Indian language)" shared task at FIRE 2025, which seeks to build a Hindi-based multilingual QA system focused on tourism in Varanasi. In contrast to conventional Information Retrieval (IR) systems that provide whole documents, contemporary QA systems use breakthroughs in Natural Language Processing (NLP) to offer precise and brief responses to user inquiries. This paper introduces a hybrid encoder-decoder model for Hindi QA that has been trained using the VATIKA dataset. A primary contribution of this research is the integration of Named Entity Recognition (NER) to improve the model's compre-

Forum for Information Retrieval Evaluation, December 17-20, 2025, Varanasi, India
*Corresponding author
prudhvisneha2003@gmail.com (P. K. R. N. Subbannagari); avanish.232it500@nitk.edu.in (A. S. Velidi); m_anandkumar@nitk.edu.in (A. K. Madasamy)

hension of input, along with an empirical assessment of its effects. This endeavor not only furthers the development of domain-specific QA for languages with limited resources but also adheres to high research standards through its rigorous methodology and evaluation.

2. Related Works

Research on Question Answering has spanned multiple paradigms, from early rule-based systems to modern deep learning frameworks. In this section, we review prior works relevant to our study, focusing on general QA architectures, multilingual challenges, encoder-decoder approaches, and the role of NER in enhancing QA models.

2.1. Overview of Question Answering Systems

Question Answering systems have undergone substantial evolution, progressing from methods based on rules and information retrieval to advanced machine learning frameworks. Contemporary QA systems are typically classified into several types, including extractive QA, where the answer is identified as a segment of text; abstractive QA, which involves generating a new answer from the context; knowledge-based QA, which queries organized knowledge bases; and systems designed for either open or closed domains. The emergence of transformer-based models [8] like BERT [1], GPT-2 [11], and BART [2] has transformed the discipline, empowering models to discern intricate linguistic structures and produce answers that are logical and contextually appropriate. These models, having been pre-trained on immense volumes of text, exhibit extraordinary abilities in understanding natural language and executing a variety of subsequent tasks, QA included.

2.2. Multilingual and Low-Resource Language QA

Creating effective QA systems for languages besides English poses distinct difficulties, particularly for those deemed low-resource due to a scarcity of digital text data or computational resources [12]. Hindi, despite its large number of speakers, frequently fits this description in the context of sophisticated NLP applications. Although universal multilingual models such as mBART [3], mT5 [13], and GPT-2 [11] are built to process various languages, their effectiveness can be less than optimal for particular low-resource languages or specialized domains. Initial explorations revealed that these general-purpose multilingual models had considerable trouble understanding Hindi context and questions, often producing text that lacked semantic coherence or relevance to the query. This suggested a basic deficiency in their capacity to genuinely acquire and apply the subtleties of Hindi for intricate tasks like question answering, rather than just generating sequences that resemble Hindi. This finding highlighted the need for a tailored approach, specifically using models that were pre-trained or fine-tuned on Hindi or other Indian languages, to obtain meaningful results. The difficulties faced with these generic models provided the justification for the specialized hybrid strategy employed in this research, which utilizes models created specifically for Indian languages like l3cube-pune/hindi-bert-v2 [14] and IndicBART [16].

2.3. Encoder-Decoder Architectures

Encoder-decoder frameworks are essential for numerous sequence-to-sequence operations in NLP [9], such as abstractive QA, machine translation, and text summarization. In this model, an encoder takes an input sequence (like a question and its context) to form a detailed contextual representation. Subsequently, a decoder utilizes this representation to produce the output sequence, such as the answer. Models like BERT [1] are particularly effective as encoders because of their bidirectional grasp of context, which makes them highly proficient at understanding input text. In contrast, models such as BART [2] and T5 [10] are excellent choices for decoders, as they can generate fluent and logical text.

Table 1
VATIKA Dataset Statistics

Split	Contexts	QA Pairs
Train	5,358	13,408
Validation	1,158	2,963
Test	1,143	2,902

The pairing of a high-performance encoder for comprehension with a dependable decoder for generation creates a powerful structure for abstractive QA.

2.4. Named Entity Recognition (NER) in QA

NER is a subfield of information extraction that focuses on identifying and categorizing named entities within text into established classes like names of people, organizations, places, times, amounts, monetary figures, and percentages [7]. Within the scope of QA, NER can act as a significant enhancement by explicitly marking key entities in the source text. This has the potential to direct the QA model's focus toward critical information, consequently enhancing its capability to either extract or formulate precise answers. For example, by classifying "Varanasi" with a `<LOCATION>` tag, the model may better grasp its significance in a query regarding tourist attractions. Recent work like Naamapadam [15] has provided large-scale named entity annotated datasets for Indian languages, enabling better NER capabilities for Hindi and other Indic languages.

3. VATIKA Dataset

To evaluate our proposed approach, we utilize the VATIKA dataset, a specialized resource curated for Hindi Question Answering in the tourism domain. This section outlines the dataset's composition, thematic scope, and statistical properties, which collectively highlight its uniqueness and relevance for low-resource NLP research.

3.1. Dataset Overview

The VATIKA dataset, introduced in the work by Gatla et al. [17], is a specially created Hindi-language Question Answering (QA) resource developed to facilitate Machine Reading Comprehension (MRC) and QA tasks related to tourism. It is centered on the culturally vibrant city of Varanasi, and its content is carefully designed to mirror authentic queries that tourists and pilgrims might have about locations, logistical information, available amenities, and spiritual sites. A key characteristic of the VATIKA dataset is its wide-ranging scope, covering ten separate domains relevant to tourism. These areas encompass Ganga Aarti, cruise services, food courts, public restrooms, Kunds, museums, general information, Ashrams, temples, and travel-related queries. Every domain in the dataset contains thorough paragraph-long contexts in Hindi, each followed by several question-answer pairs. This arrangement is intended to replicate real-life information-seeking scenarios using natural language, offering a comprehensive and diverse platform for testing QA systems. The questions vary widely, from those seeking factual information to inquiries about navigation and experiences, thereby covering a broad spectrum of common tourist questions. The entire VATIKA dataset is in Hindi, written in the Devanagari script, which makes it an important linguistic tool for creating and assessing QA systems designed for Indian languages. It is suitable for both open-domain and context-based MRC-style question answering, rendering it adaptable for a range of research applications.

3.2. Dataset Statistics

The VATIKA dataset is divided into separate subsets for the purposes of model training, validation, and testing. Table 1 provides a breakdown of the distribution of contexts and QA pairs among these divisions.

For this research, the initial training and validation collections were merged to create a more extensive training dataset. A segment of the official test data was subsequently used as a validation set during the stages of model development and parameter adjustment. Although this method expands the amount of data for training, it is crucial to recognize that using a portion of the test set for validation could result in an overly positive assessment of the model's ability to generalize to the complete, unseen test set. This is due to the possibility that the model may indirectly adapt to certain features of the final evaluation data. Consequently, the final test results presented should be viewed in consideration of this particular data allocation method.

4. Methodology

Our methodology integrates a hybrid encoder–decoder framework with additional NER-based enhancements. This section describes the problem formulation, training configuration, architectural choices, and the mechanisms used for answer generation. The design rationale behind each component is also discussed.

4.1. Problem Formulation

The task of Question Answering is framed as a problem of sequence-to-sequence conversion. The model receives an input that is a concatenated sequence of the user's question and the associated context document, with both elements presented in Hindi. The model's goal is to produce a natural language answer sequence in Hindi that provides a direct response to the question, based on the given context.

4.2. Training Configuration

The model's training was carried out with the AdamW optimizer [4], set up with a learning rate of 3×10^{-5} and a weight decay of 0.01. To properly regulate the learning rate during training, a linear schedule with a warmup phase was used. The total number of training steps was determined based on running for 5 epochs over the consolidated training data. For effective training, the model was run on a computational device equipped with a GPU.

4.3. Proposed Hybrid Model (BERT-Encoder, IndicBART-Decoder)

4.3.1. Data Preparation (QADataset)

The QADataset class handles the preparation of input and target data for the hybrid model. The encoder input is created by concatenating the question and context into a single string with the format "question: [question text] context: [context text]". This merged text is subsequently tokenized with the l3cube-pune/hindi-bert-v2 [14] tokenizer. To maintain consistent input sizes, sequences were padded or truncated to a maximum length of 512 tokens. For the decoder's target, the correct answer is tokenized via the ai4bharat/IndicBART [16] tokenizer, with a maximum length of 128 tokens. A key part of this procedure is the adjustment of special tokens. The BERT tokenizer employs `[CLS]` and `[SEP]` tokens, but IndicBART needs `<s>` (start of sentence) and `</s>` (end of sentence) tokens for generation. Consequently, the token ID for `[CLS]` from BERT's vocabulary is substituted with IndicBART's `<s>` ID, and the `[SEP]` token ID is swapped with the `</s>` ID in the target sequences. Additionally, padding tokens within the target sequence are masked by changing their IDs to -100. This masking guarantees that the loss function overlooks these padding tokens during backpropagation, concentrating only on

the actual content of the answer. These apparently minor adjustments are essential for the model's capacity to learn and produce logical answers, reflecting a thorough strategy for data preparation and the alignment of different model parts.

4.3.2. Encoder-Decoder Integration

A vital feature of this hybrid design is the smooth connection between the BERT encoder and the IndicBART decoder. The l3cube-pune/hindi-bert-v2 [14] model produces hidden states with a dimension of 768, whereas the ai4bharat/IndicBART [16] model requires an input dimension of 1024 for its encoder outputs. To resolve this dimensional inconsistency and facilitate efficient information flow, a linear projection layer was integrated. This layer performs a learnable transformation, converting the 768-dimensional hidden states from BERT into the 1024-dimensional format that IndicBART anticipates. This design decision exemplifies a refined method for merging disparate pre-trained models, underlining the adaptability needed when building custom architectures from components trained separately. The effectiveness of this projection layer can heavily impact the model's overall performance, marking it as a candidate for additional refinement, such as investigating more intricate non-linear transformations.

The hybrid BERT-IndicBART architecture is illustrated in Figure 1.

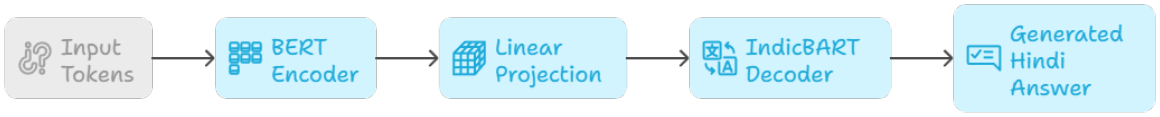

Figure 1: Hybrid BERT-IndicBART architecture showing the encoder-decoder integration with linear projection layer.

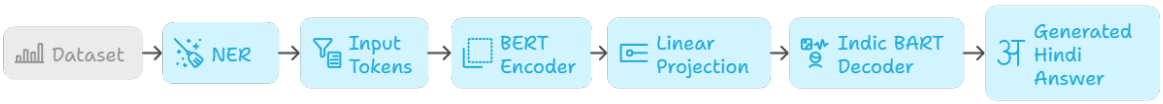

Figure 2: A hybrid BERT-IndicBART model integrates an encoder-decoder with a linear projection layer, using NER-processed data.

4.3.3. Answer Generation

To produce answers, the generate method of the BertIndicBARTQA model was employed, which uses a beam search with four beams. This approach aids in considering multiple possible answer sequences to identify a better one, instead of just choosing the most likely token at every stage. A number of parameters were fine-tuned empirically to improve the quality and smoothness of the generated answers. These included setting a maximum of 128 new tokens to cap answer length, enforcing a minimum length of five tokens to prevent overly short responses, and enabling early stopping to halt generation once a complete sequence is found. To further improve quality, a repetition penalty of 1.2 was applied to deter repetitive phrasing, a length penalty of 1.0 was used to avoid favoring overly long or short answers, and recurring trigrams were prevented to enhance fluency. Moreover, specific bad word IDs were set for the `[CLS]` and `[SEP]` tokens to stop them from being generated, making sure the output conforms to natural language standards. The decoder start token ID was dynamically configured to the

`hi_IN` language code ID for IndicBART, guaranteeing the correct start for language-specific generation. This meticulousness in the generation plan indicates a sophisticated approach to abstractive QA, where the quality of the generated text is considered as crucial as the model's underlying understanding.

4.4. NER-Enhanced Model

4.4.1. NER Integration Strategy

To explore the possible advantages of supplying direct semantic indicators, a second model version was created that integrates NER into the input text. The goal of this method was to emphasize key entities, like locations or organizations, within the input sequence itself, thereby directing the focus of the QA model. The ai4bharat/IndicNER [15] tokenizer and model were utilized for this task.

4.4.2. NER Processing

The integration was carried out using two primary functions. The first, `get_ner`, tokenizes an input sentence, retrieves the predicted entity categories from the ai4bharat/IndicNER model, and then matches these entity labels to the original words in the sentence as shown in Figure 2. This results in a list of `(word, entity_label)` pairs. The second function, `enhance_with_ner`, then handles this output. For any word recognized as a named entity (i.e., not having the "O" label for "Other"), it embeds distinct XML-style tags around the word to show its entity category. As an illustration, if "Varanasi" is recognized as a location, it would be transformed to `<LOCATION>Varanasi</LOCATION>`. This technique of adding explicit tags directly into the input text is a frequently used method to give models structural information.

4.4.3. Integration with QA Dataset

This NER-based modification was implemented within the `QADataset.__getitem__` method. The raw combined question and context string (`input_text_raw`) was first processed by the `enhance_with_ner` function. The modified string (`input_text`) was subsequently used as the input for tokenization by the l3cube-pune/hindi-bert-v2 tokenizer. Although this approach seems logical, it can create difficulties, as the new tags become unfamiliar tokens that the BERT tokenizer, pre-trained on plain text, might not process semantically. This could potentially interfere with original tokenization patterns or result in the tags being interpreted as random noise.

5. Experiments and Results

We conducted a series of experiments to assess the effectiveness of our proposed hybrid model, as well as its NER-augmented variant. This section details the experimental setup, evaluation metrics, and a comparative performance analysis against baseline models.

5.1. Experimental Setup

The experiments utilized the l3cube-pune/hindi-bert-v2 [14] tokenizer for processing inputs and the ai4bharat/IndicBART [16] tokenizer for generating target outputs. Training was conducted for 5 epochs with the AdamW optimizer [4], which was set to a learning rate of 3×10^{-5} and a weight decay of 0.01. A linear learning rate schedule that included warmup steps was also implemented. To guarantee swift training and inference, all computational tasks were executed within a GPU-accelerated environment.

5.2. Evaluation Metrics

A comprehensive set of established evaluation metrics was used to rigorously measure the performance of the QA models:

Exact Match (EM): This metric measures the percentage of predictions that precisely match the ground truth answer. It is a strict measure, requiring character-level identity.

F1 Score: The harmonic mean of precision and recall, the F1 score is particularly useful for evaluating tasks where partial matches are acceptable. It provides a balanced measure of a model's ability to retrieve relevant information and avoid irrelevant information.

BLEU (Bilingual Evaluation Understudy): BLEU [5] is a widely used metric for evaluating the quality of text generated by a machine translation system, but it is also applicable to abstractive QA. It quantifies the n-gram overlap between the generated answer and a set of reference answers, indicating fluency and adequacy. BLEU-1 considers unigram overlap, while BLEU-2 considers bigram overlap.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation): ROUGE [6] is a set of metrics commonly used for evaluating summarization and machine translation. It measures the overlap of n-grams, word sequences, and word pairs between the generated answer and reference answers. ROUGE-1 measures unigram overlap, ROUGE-2 measures bigram overlap, and ROUGE-L measures the longest common subsequence, reflecting sentence-level structure and fluency.

5.3. Performance Analysis

The performance of both the base BERT-IndicBART hybrid model and the NER-enhanced variant was evaluated on the test set. The results are summarized in Table 2 and Table 3.

Table 2
Model Performance Metrics on Validation Dataset

Metric	BERT + BART	BERT + BART + NER	google/mt5-base
Exact Match	57.6%	46.3%	44.0%
Average F1 Score	0.9092	0.8710	0.7966
BLEU-1	0.8543	0.7994	0.8097
BLEU-2	0.8203	0.7565	0.5109
ROUGE-1	0.2130	0.1996	0.1867
ROUGE-2	0.1173	0.1016	0.9846
ROUGE-L	0.2122	0.1985	0.1879

Table 3
Models performances on the test dataset of FIRE during evaluation

Metric	BERT + BART	BERT + BART + NER	google/mt5-base
BLEU-1	52.3	46.5	63.7
BLEU-2	25.9	21.8	41.2
BLEU-3	14	11	29.4
BLEU-4	8.1	5.9	22.5
ROUGE-1	0.0288	0.0201	0.0561
ROUGE-2	0.0154	0.0094	0.0260
ROUGE-L	0.0276	0.0192	0.0561
QA-F1	0.394	0.352	0.505

The results reveal several important findings. The base BERT-IndicBART hybrid model achieved strong performance on the validation set, with an exact match score of 57.6% and an F1 score of 0.9092.

Notably, the addition of NER tags consistently decreased performance across all evaluation metrics. This counterintuitive result warrants a deeper analysis. A critical factor was a practical limitation imposed by the NER integration strategy. The process of inserting XML-style tags to denote named entities nearly **doubled the length of the input sequences**. This substantial increase in data size meant that, given the available computational resources and time constraints, the NER-enhanced model could not be trained for the same number of epochs as the baseline model.

Consequently, the primary reason for the diminished scores is most likely that the model was **severely undertrained**. An undertrained model would not have had sufficient exposure to the data to learn the complex patterns of the QA task, nor could it effectively learn to interpret the novel syntax introduced by the NER tags. This issue was likely compounded by secondary factors, such as the tokenizer splitting the unfamiliar tags into noisy subwords, creating an even more challenging learning environment for a model that was already not receiving adequate training. Therefore, the combination of a more complex input representation and an unavoidable reduction in training duration ultimately led to the observed degradation in performance.

Furthermore, it is noteworthy that the `google/mt5-base` model secured the highest performance on the official FIRE test set, achieving the top rank among all participants across all four BLEU scores.

6. Conclusion

This research led to the successful creation and assessment of a hybrid BERT-IndicBART Question Answering system tailored for the VATIKA dataset, proving its proficiency in delivering precise Hindi answers to tourism-focused questions about Varanasi. The fundamental hybrid model, which paired the l3cube-pune/hindi-bert-v2 model as an encoder with the ai4bharat/IndicBART model as a decoder, yielded robust results across crucial metrics like Exact Match, F1 Score, BLEU, and ROUGE, demonstrating its competence in managing domain-specific Hindi QA.

A surprising and significant discovery from this work was the decline in performance that occurred when Named Entity Recognition (NER) tags were inserted directly into the input text. Although the NER modification was designed to offer clear semantic pointers to the model, it invariably resulted in lower scores on every evaluation metric. This result points to the inherent difficulties in feature interaction and representation when merging external knowledge with pre-trained language models, especially within the framework of low-resource languages such as Hindi. It emphasizes that the method of presenting information to a model can be just as crucial as the information itself.

Acknowledgments

The authors would like to thank the organizers of the FIRE 2025 VATIKA shared task for providing the dataset and evaluation framework. We also acknowledge the computational resources provided by our institutions that made this research possible.

Declaration on Generative AI

The author's have not employed any Generative AI tools in the preparation of this work.

References

[1] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," *Proc. NAACL-HLT*, pp. 4171–4186, 2019.

[2] M. Lewis, Y. Liu, N. Goyal, et al., "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," *Proc. ACL*, pp. 7871–7880, 2020.

[3] Y. Liu, M. Ott, N. Goyal, et al., "Multilingual Denoising Pre-training for Neural Machine Translation," *Trans. ACL*, vol. 8, pp. 726–742, 2020.

[4] I. Loschilov and F. Hutter, "Decoupled Weight Decay Regularization," *Proc. ICLR*, 2019.

[5] K. Papineni, S. Roukos, T. Ward, and W. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation," *Proc. ACL*, pp. 311–318, 2002.

[6] C.-Y. Lin, "ROUGE: A Package for Automatic Evaluation of Summaries," *Text Summarization Branches Out: Proc. ACL Workshop*, pp. 74–81, 2004.

[7] D. Nadeau and S. Sekine, "A Survey of Named Entity Recognition and Classification," *Lingvisticae Investigationes*, vol. 30, no. 1, pp. 3–26, 2007.
[8] A. Vaswani et al., "Attention Is All You Need," *Proc. NIPS*, pp. 5998–6008, 2017.
[9] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," *Proc. NIPS*, pp. 3104–3112, 2014.
[10] C. Raffel et al., "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer," *J. Mach. Learn. Res.*, vol. 21, no. 140, pp. 1–67, 2020.
[11] A. Radford et al., "Language Models are Unsupervised Multitask Learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.
[12] N. Arivazhagan et al., "Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges," *arXiv preprint arXiv:1907.05019*, 2019.
[13] L. Xue et al., "mT5: A Massively Multilingual Pre-trained Text-to-Text Transformer," *Proc. NAACL-HLT*, pp. 483–498, 2021.
[14] R. Joshi, "L3Cube-HindBERT and DevBERT: Pre-Trained BERT Transformer models for Devanagari based Hindi and Marathi Languages," *arXiv preprint arXiv:2211.11418*, 2022.
[15] A. Mhaske, H. Kedia, S. Doddapaneni, et al., "Naamapadam: A Large-Scale Named Entity Annotated Data for Indic Languages," *arXiv preprint arXiv:2212.10168*, 2022.
[16] R. Dabre et al., "IndicBART: A Pre-trained Model for Natural Language Generation in 11 Indian Languages," *Proc. EMNLP*, pp. 7479–7494, 2022.
[17] P. Gatla, Anushka, N. Kanwar, G. Sahoo, and R. K. Mundotiya, "Tourism Question Answer System in Indian Language using Domain-Adapted Foundation Models," *arXiv preprint*, 2025.