International Journal of Technology and Applied Science (IJTAS)

E-ISSN: 2230-9004 ● Website: www.ijtas.com ● Email: editor@ijtas.com

Multilingual NLP Systems for Low-Resource Languages

Mrs. K. Swapna[1], P. Riya[2], B. Hari Priya[3], D. Sangeetha[4], N. Sowmya[5]

[1]Assistant Professor, [2,3,4,5]B. Tech 3rd Year Student

[1,2,3,4,5]CSE(AI&ML), Vignan's Institute of Management and Technology for Women, Hyderabad, India.

Abstract:

India is home to many different languages, but some are not getting much attention. These languages have few digital records, not many tagged texts, and not much help from technology. Working on natural language processing (NLP) in India is tough because of the many different ways people speak, the complex rules of each language, and the lack of standard benchmarks. To deal with this, systems are being developed that are not one-size-fits-all but are tailored to fit the needs of these overlooked languages. These systems quietly help close the gap without making a big deal about it. One way to handle this is by learning from languages that have a lot of data, such as Hindi, English, or Tamil, and then applying that knowledge to languages with less data. Instead of starting from scratch for each language, models can borrow knowledge from one language to improve another. Techniques like transfer learning, word maps that span multiple languages, and network designs built for handling different contexts are being tested when there's not a lot of data available. Multilingual models perform much better than single-language models when it comes to languages with limited data [3] Transfer learning and cross-lingual embeddings let models share knowledge across languages. Semi-supervised and unsupervised methods help use limited labeled data more effectively. These approaches greatly improve NLP performance for low-resource Indian languages and support digital inclusion. With no need for endless labeled examples, the hidden structures in raw text start to show up thanks to unguided training styles. Even partial labels can help nudge the system forward under half-guided setups. What's special here is how it brings together traits that are common in Indian languages, like word structure and different scripts, because these are often found in everyday conversations. These models are tested across various language tasks, such as translating speech from one language to another, detecting emotions in text, and extracting names from sentences. All these tasks are aimed at checking how well the models adapt while maintaining strong performance. It turns out multilingual models work much better than single-language models for languages with little data. [4] What stands out is how working together on datasets, sharing tools freely, and involving local communities can make a real difference for less common languages in natural language processing.

I. INTRODUCTION:

India ranks among the top linguistically diverse countries in the world with many languages spoken in different regions. However, while major Indian languages like Hindi, English, and Tamil have witnessed major leaps in Natural Language Processing (NLP) techniques, many other Indian languages have not yet found their rightful place in the digital world. These languages have been referred to as low-resource languages because they do not have enough data, linguistic support, and technological infrastructure. Natural Language Processing is an integral part of modern technology, and its application can be seen in machine translation systems, speech recognition systems, chatbots, sentiment analysis tools, and many more. However, most Natural Language Processing techniques have been developed for high-resource languages, and it is not easy to apply these techniques to other Indian languages because there is not enough data and documentation on these languages. These languages have been referred to as low-resource languages because they do not have enough data, linguistic support, and technological

infrastructure. Most NLP techniques have been developed for high-resource languages, making adaptation difficult for Indian language. Multilingual NLP systems allow knowledge transfer from resource-rich languages to low-resource languages [8] improving performance in tasks like translation and sentiment analysis [7]

There is also a lack of uniformity in scripts, many complex morphological features, and code-mixing in Indian languages. For this purpose, multilingual NLP systems have been proposed as a promising solution to handle such difficulties. These systems can efficiently learn from multiple languages simultaneously and can also efficiently transfer knowledge from a resource-rich language to a low-resource one. The main aim of this research is to develop and analyze various multilingual NLP approaches for low-resource Indian languages. With the help of linguistic similarities and recent advances in ML, this research aims to develop efficient and inclusive systems for low-resource Indian languages. This research can significantly contribute to bridging the technological gap and promoting digital inclusion for low-resource Indian languages, thereby preserving India's linguistic diversity. To overcome these difficulties, multilingual NLP systems have been proposed as a promising solution. Multilingual NLP systems can learn multiple languages simultaneously and can also share knowledge across languages, particularly from a resource-rich to a low-resource language. This research aims to develop and analyze multilingual approaches to NLP for low-resource Indian languages. With the help of linguistic similarities and recent advances in machine learning, it is possible to develop efficient and inclusive NLP systems for low-resource Indian languages. This research aims to bridge the technological gap and promote digital inclusion in India, preserving its linguistic diversity.

II. RELATED WORK:

Research work carried out in multilingual Natural Language Processing (NLP) for low-resource Indian languages has attracted much more attention in recent times, mainly because of the increased need for language technologies. In this regard, it is noted from existing studies that the majority of the advancements reported for NLP have been favorable for high-resource language development, and many Indian languages have not received the required importance. This led the scientific community to look for alternative solutions that can work well even for low-resource languages. One of the prominent areas of existing work is the application of multilingual and cross-lingual learning. This type of learning is mainly focused on utilizing the knowledge of high-resource languages, like English or Hindi, for improving the performance of low-resource languages. Research studies have reported that multilingual pre-trained models can greatly improve question answering and translation tasks by utilizing the common patterns of language. In addition, a cross-lingual transfer learning framework is proposed for better language generalization and improving the performance of low-resource languages. Another prominent field of research includes data augmentation and corpus development. In light of the scarcity of data, it is a significant challenge. Researchers have been focusing on developing parallel corpora, thereby improving the data. For instance, a recent study of Indian language machine translation introduced parallel corpora for under-resourced languages like Bhili, thereby showing the importance of data development for improving the overall performance of machine translation. Data collection by the community and open-source development have been highlighted as a prominent solution for this problem. Moreover, transformer-based models like mBART, mT5, and IndicBART have been extensively used for tasks like text summarization and translation for Indian language development. Multilingual and cross-lingual learning approaches have shown significant improvements in low-resource NLP tasks . Pre-trained multilingual models improve translation and question answering tasks. Data augmentation techniques and corpus development are essential to overcome data scarcity. Transformer-based models such as mBART and mT5 have been widely used for Indian language tasks.

Cross-lingual transfer learning frameworks enhance generalization across languages, while community-driven data collection improves dataset availability Linguistic challenges such as morphological

complexity and code-mixing require specialized preprocessing techniques [2]
Multilingual and cross-lingual learning approaches have shown significant improvements in low-resource NLP task. Pre-trained multilingual models improve translation and question answering tasks. Data augmentation techniques and corpus development are essential to overcome data scarcity. Transformer-based models such as mBART and mT5 have been widely used for Indian language tasks [5]

Cross-lingual transfer learning frameworks enhance generalization across languages, while community-driven data collection improves dataset availability. Linguistic challenges such as morphological complexity and code-mixing require specialized preprocessing techniques.

The overall performance of the model is satisfactory, although it is limited by the quality of the data. The overall performance of the model can be enhanced by combining multiple related languages for tasks like Named Entity Recognition (NER), especially when there are more similarities between the languages, like Hindi and Marathi. Other research focuses on linguistic issues unique to Indian languages, such as their morphological complexity, script variety, and code-mixing. Research on tokenization and preprocessing techniques has shown that current NLP techniques may not effectively handle such unique aspects of Indian languages and may require specific adaptations for better results. Another important finding is recent advancements in text summarization tasks, which show that low-resource Indian languages still require standardized evaluation metrics to progress. From the above research, it is evident that a considerable amount of progress has been made in low-resource Indian languages through multilingual models, transfer learning, and data augmentation techniques. However, there are still many limitations to be resolved before a more robust and scalable NLP system can be developed for low-resource Indian languages.

III. PROPOSED SYSTEM:

The proposed system is based on the development of an efficient and scalable multilingual natural language processing system that is tailored for low-resource Indian languages. Unlike the existing systems that are heavily dependent on high-resource languages, the proposed system tries to bridge the linguistic gap with the implementation of advanced features such as transfer learning and data augmentation. The fundamental idea of the proposed system is based on the implementation of the pre-trained multilingual transformer-based approach that helps in the better comprehension and processing of multiple languages. The implementation of such models that are trained on various linguistic datasets helps in the sharing of knowledge. The system tries to utilize this knowledge and implement the generalization of the linguistic features of high-resource languages and apply them to low-resource languages. In order to address the problem of data scarcity, the system utilizes cross-lingual learning. Using this approach, the system can learn the relationships that exist between different languages. This helps the system perform better in tasks such as translation, sentiment analysis, and entity recognition. The system also utilizes data augmentation techniques such as back-translation. In this technique, the sentences in the low-resource language are translated into another language and then back again. The proposed system uses multilingual transformer-based models to improve language understanding [4] Cross-lingual learning enables knowledge transfer between languages, while data augmentation techniques like back-translation improve robustness. Language-specific fine-tuning enhances performance for individual languages [3]

This helps the system become more robust. Another important feature that the proposed system offers is language-specific fine-tuning. Although the multilingual models are robust, they do not perform well in all languages. The problem with the multilingual models is that they do not recognize the unique grammatical structures and morphological features that each language offers. In order to address this problem, the system offers the facility of fine-tuning the models with the data that is available in each language. It was designed to expand in the future, but for now, it follows several phases. One process leads

to the next, such as gathering information, then moulding it, followed by teaching the model. Cleaning is an early process, which raises the quality of what is used. Information is broken into pieces, standardized, and unnecessary items removed. Instead of creating something new, an existing model is used but modified, including adding new functions in understanding speech and text. After that, testing occurs, followed by going live. To determine if the setup works well, several measures are used. One of the main reasons for creating this system was to reduce the heavy computational requirements that usually accompany standard language processing techniques. Because it utilizes existing models, making changes for various functions is easier. This means less strain on the computer, which can benefit areas with limited support. It drives chatbots, enabling them to comprehend various tongues. When people talk in different ways, converting speech to text facilitates things. When voice tools respond, they can use various tongues to do so. Information exchange through language barriers appears to be less of a challenge. An individual speaking in Hindi can communicate with another speaking in Tamil. Language barriers are crossed with a whisper. Grammar books failed to bridge such language gaps, but rhythm does. Immediately, it dives into the challenges of less common Indian tongues in tech-related language exercises. It does not wait in idle mode; instead, it relies on interconnected models that traverse language barriers, enhancing comprehension and translation capacities. It is designed to improve, with more speech forms joining in the future without any hiccups. Further, improved data pools fill in the gaps where vocabulary is limited. Learning from dominant tongues enables fewer common tongues to grow at a rapid pace. Expanding digital presence is the outcome of such language exchange. Tech assists troubled speech with unseen support. Voice features can be integrated in the future without any issues. Systems that transform, such as this one, make everyone feel they belong. Back doors transform into front doors with such smooth changes.

V. METHODOLOGY

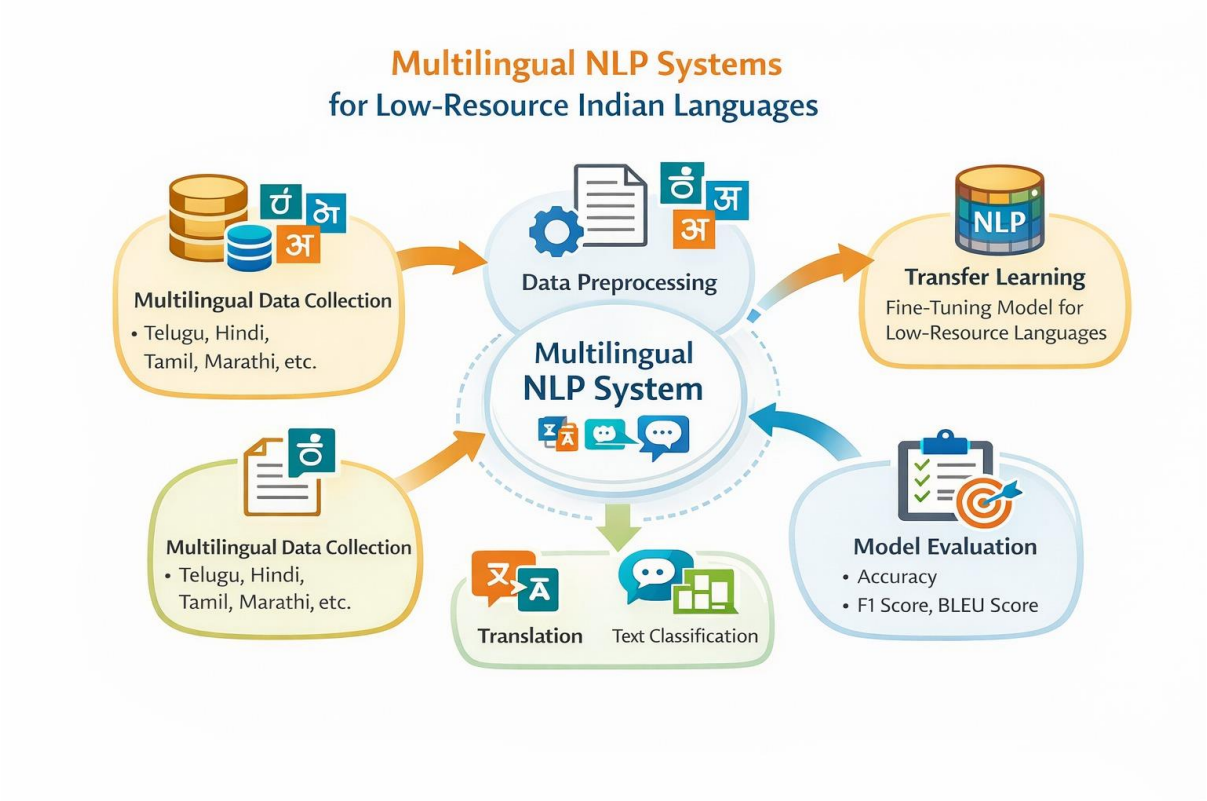
One way this work moves forward is by building a strong multilingual tool for Indian languages with little digital support, shaping it carefully over time. Starting off, specific tongues are picked along with the jobs they must handle through natural language processing. Languages such as Bodo, Santali, and Tulu stand out here - these often get overlooked due to their small footprint online. What helps most is focusing energy where data runs thin. Once the languages and natural language processing goals are set, gathering information comes next. Because those languages often lack large amounts of digital material, content must come from scattered places - archives online, public records, official pages, published writings, among others. When needed, paired texts in multiple tongues are gathered too, especially useful for certain types of analysis. The quality of what's pulled together here shapes how well systems later perform - it quietly sets limits or opens doors without announcing itself. What feeds the process at this stage leaves a clear mark downstream. Once the data is collected, it is then refined in order to make it more workable. Transfer learning is applied using pre-trained multilingual models to reduce computational cost. Cross-lingual learning improves low-resource language performance by leveraging high-resource languages. Data augmentation techniques such as back-translation increase dataset size and diversity. Evaluation metrics such as precision, recall, and F1-score are used to measure performance [1]

Cleaning is done by removing unusual symbols, correcting spelling errors, and standardizing text. After that, sentences are broken down into smaller parts, either words or parts of words. This is known as tokenization. When dealing with Indian languages, text encoding is also crucial. Each of these processes is aimed at ensuring that the data is in good condition in order for learning to occur effectively. Failure to do so may result in unpredictable outcomes. Therefore, preprocessing is significant in enhancing the quality of learning. Before any other thing is done, a pre-trained model is selected. Most of these models have been developed using the transformer model. This is because they have been trained on more than one language. Therefore, they have been exposed to more than one language. It is not necessary to train a model from scratch. This is because it requires more time and computational power. However, it is

possible to apply transfer learning. A model is fine-tuned using a particular data set meant for low-resource languages. This is aimed at enhancing its knowledge in order to perform better in particular language patterns. For further improvement in the performance of the model, cross-lingual learning methods are used. In this approach, the knowledge of the high-resource languages, such as Hindi and English, is used to improve the low-resource languages. In addition to this, data augmentation methods such as back-translation are also used to artificially increase the data. For instance, the sentences in the low-resource language are translated into another language and then back-translated. This helps the model learn better and become robust. After the training process is complete, the model is tested and evaluated. The evaluation is done based on the accuracy, precision, recall, and F1 score. The evaluation helps in understanding the performance of the system and the improvements that need to be made.

VI. SYSTEM ARCHITECTURE:

A design choice shapes how well small-data Indian tongues get handled - features plug into place piece by piece. Built like a chain of separate blocks, every part acts on its own task without waiting. One follows another, yet each moves independently through assigned steps. From the very start, the setup begins with gathering raw information through a data entry stage. Text pulls come from places like web pages, local dialect records, official files, among others. One part handles single-language content, while another manages multiple languages at once. Spoken words go in, written lines too - both fit into how it operates. Built right, it adapts without fuss to different language needs**.** After that comes the data preprocessing step. It gets unrefined information ready for what follows. Out go distortions in the words - things like odd symbols, misplaced marks, unrelated bits. Cleaning happens here so nothing messy moves forward. At this stage, the system adjusts spelling, splits words, then assigns codes. Handling how Indian languages are written falls here too, along with their unique rules and word forms. After cleaning the data, it moves to a stage focused on expansion across languages. That piece handles shortages by boosting available examples. Instead of just using raw inputs, methods like translating back and forth come into play. Knowledge shifts from tongues with plenty of material - say, English or Hindi - to those lacking resources. The process leans on shared patterns between speech systems to fill gaps. Inside lies a level designed to process multiple languages together. Resting on prior training done with transformer models. Speech arrives here mixed across different dialects simultaneously. Learning shifts afterward through insights borrowed from past exercises.



VII. IMPLEMENTATION:

A group of machine learning tools is used as the core of the multilingual NLP environment. It is built in stages, with each stage being precise yet flexible in its potential for improvement. Coding frameworks are used to connect with intelligent ways of structuring data. All of these are easy to understand and follow.

First, we need to prepare the coding environment. Python is used because it is good with text and machine learning. TensorFlow or PyTorch is used to build and train models. For working with words, sentences, and meanings, we have resources such as Hugging Face Transformers and NLTK. The next step is to collect data and store it securely. We collect words from small-language areas of India from local sites, data sets, or mixed-language data sets found online.

After collecting it, we organize it in neat formats such as CSV or JSON files. Pre-trained models like mBERT and XLM-R are used for multilingual NLP tasks [6] Transfer learning enables efficient model training with limited data [1] Back-translation is used for data augmentation to improve robustness. Tools such as Hugging Face Transformers support implementation of these models.

Sometimes we also use live data or web pages to collect data in case we need more of it. After that comes cleaning the data. Removing extra symbols happens here, while gaps in information get handled too. Sentences split into pieces during this phase. Breaking words into smaller units follows, especially useful for Indian languages. Such methods make sure inputs fit well for learning later. Getting things right at this stage shapes what follows. One way to start is by using a model already trained on many languages. Examples like mBERT or XLM-R come into play here.

Tools such as Hugging Face Transformers help bring these models to life. Instead of building something entirely new, there's value in reusing what exists. Training happens on data gathered specifically for less common languages. That approach? It goes by the name of transfer learning. When there's not a lot of data, people resort to tricks. Back-translation sometimes pops up. A sentence is translated into another language and then translated back. It ends up a bit different. New versions are created this way. More

versions mean that the models learn how to adapt better later on. A setup consists of a series of steps that are taken for training and testing. The steps start by dividing the data into a section for practice and a section for testing later.

The practice section requires appropriate settings, such as how fast to learn, how many examples to use, and how many rounds of learning. Once the learning is done, testing begins, and numbers are used to display how well the model guesses, catches, or incorrectly catches. The numbers provide a clear picture of how the model works with new information. A simple front-end for applications sometimes pops up. It appears when there's a need for it to be live. A simple example is typing words that are translated from one language into another or checking if words sound happy or sad.

It sometimes runs online or runs from typing out words on a screen. It simply works and does what it needs to. Once everything is ready, the setup begins on a computer or an online host, allowing people to access it. To make it go faster and use less power, changes are made while it is running. This is because the pieces fit together in a block-like fashion, making it easy to insert different tools or languages later.
It works from beginning to end because it is designed to fill real-world needs—performing well even when there are large gaps in data for smaller Indian languages, which is where it is most helpful and makes NLP accessible in those regions.

VIII. ALGORITHM:

Step 1: Start and Input Acquisition
- Initialize the system and accept user input through interface

Step 2: Language Detection
- Identify the source language
- Handle unknown or mixed-language input
- Assign language label for processing

Step 3: Text Preprocessing
- Remove noise (special characters, extra spaces)
- Normalize text (lowercasing, script standardization)

Step 4: Tokenization
- Split text into tokens (words/sub words)
- Use multilingual tokenizer
- Prepare tokens for model input

Step 5: Embedding Generation
- Convert tokens into vector representations
- Use pre-trained multilingual embeddings
- Capture contextual meaning of text

Step 6: Cross-Lingual Transfer
- Map embeddings into shared semantic space
- Transfer knowledge from high-resource languages
- Improve representation of low-resource inputs

Step 7: Task Processing
- Select required NLP task (translation / sentiment / summarization)

- Apply task-specific model
- Generate intermediate results

Step 8: Output Generation and End
- Convert processed data into readable text
- Perform post-processing (formatting, grammar correction)
- Display final output to user and terminate system

IX. DISCUSSION:

Right now, computers handle big languages like English or Hindi much better when dealing with spoken or written words. Yet tiny ones - Tulu, Bodo, Santali - lack enough online text to work with. Without any digital traces, tools simply do not exist for people using these tongues daily. Imagine typing in your mother tongue only to meet silence from the machine. This challenge drives our work - building ways for machines to grasp meaning, no matter how brief the message. Voices stay silent when systems fail them; fixing that imbalance guides every step forward. Multilingual models significantly improve performance for low-resource languages when fine-tuned properly [9].Data scarcity remains a major challenge, requiring augmentation and transfer learning techniques. Cross-lingual approaches enable better generalization and improved accuracy [10]

Getting enough good data posed a major hurdle across the whole effort. While English had more than enough material, many other tongues came with almost nothing useful. The scraps that did exist were often messy or unreliable. Training fresh systems straight through turned out nearly impossible under those conditions. Existing models built for multiple languages offered a path forward instead. Knowledge gained from one language helped lift another into reach.

It's been discovered that multilingual systems work much better once adjusted for individual tongues. Of course, at first, certain elements such as word order or meaning could be entirely overlooked. However, with even a small amount of training, accuracy is definitely improved. The key is how that information is being shaped during the process

X. RESULT:

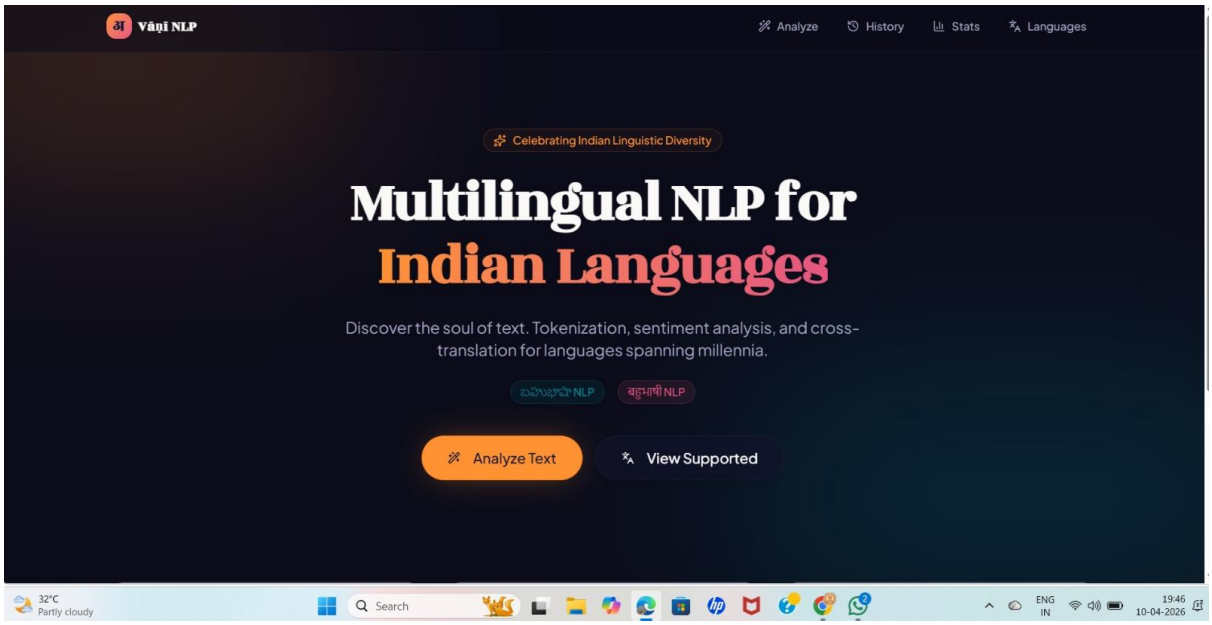


Fig 9.1. Opening interface of the website

The starting interface of the proposed multilingual NLP system is designed to provide a simple, user-friendly, and visually appealing experience. The homepage presents a clean layout with culturally inspired elements representing the diversity of Indian languages. Users are welcomed with an intuitive input section where they can enter text in any supported language.

Additionally, the design incorporates modern UI components such as responsive input fields, navigation menus, and visually engaging graphics to enhance usability. The system ensures smooth interaction by guiding users toward selecting their desired NLP task, such as translation, sentiment analysis, or summarization. Overall, the starting interface acts as an entry point that combines functionality with aesthetic design, encouraging users to interact with the multilingual system.

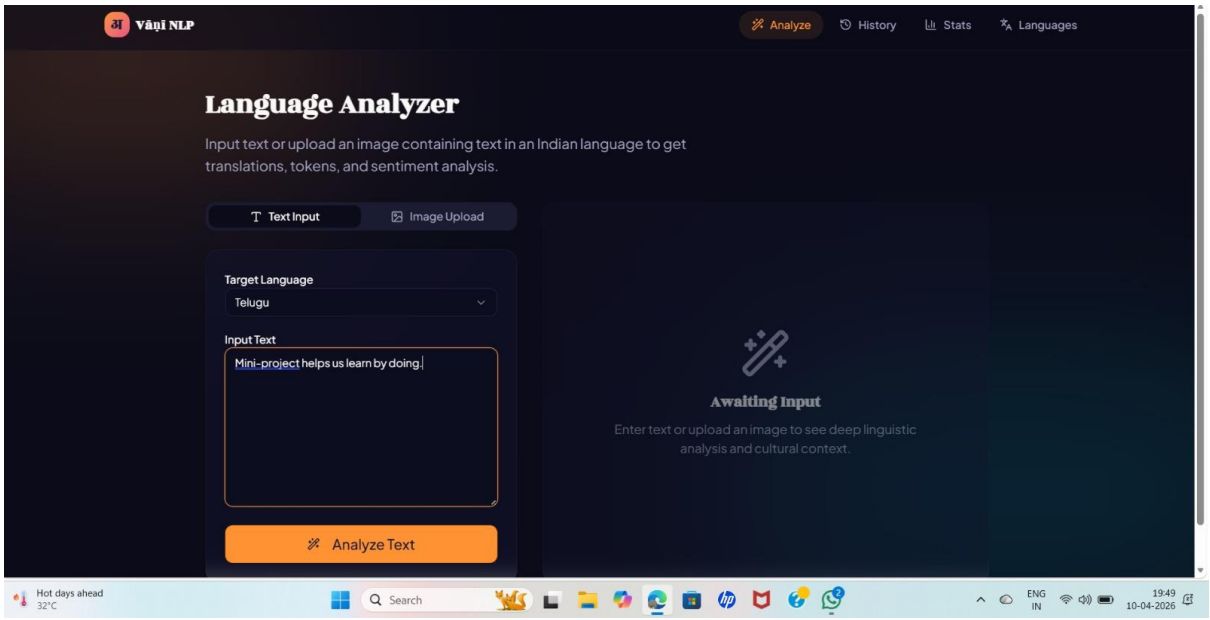


Fig 9.2. Input page of the website

The input page of the system is designed to provide a flexible and interactive environment for users to enter text and select desired processing options. It includes a clearly visible text input area where users can type or paste content in any supported Indian language. The interface also offers multiple task options such as translation, text scanning, and combined processing, allowing users to perform single or multiple operations simultaneously.

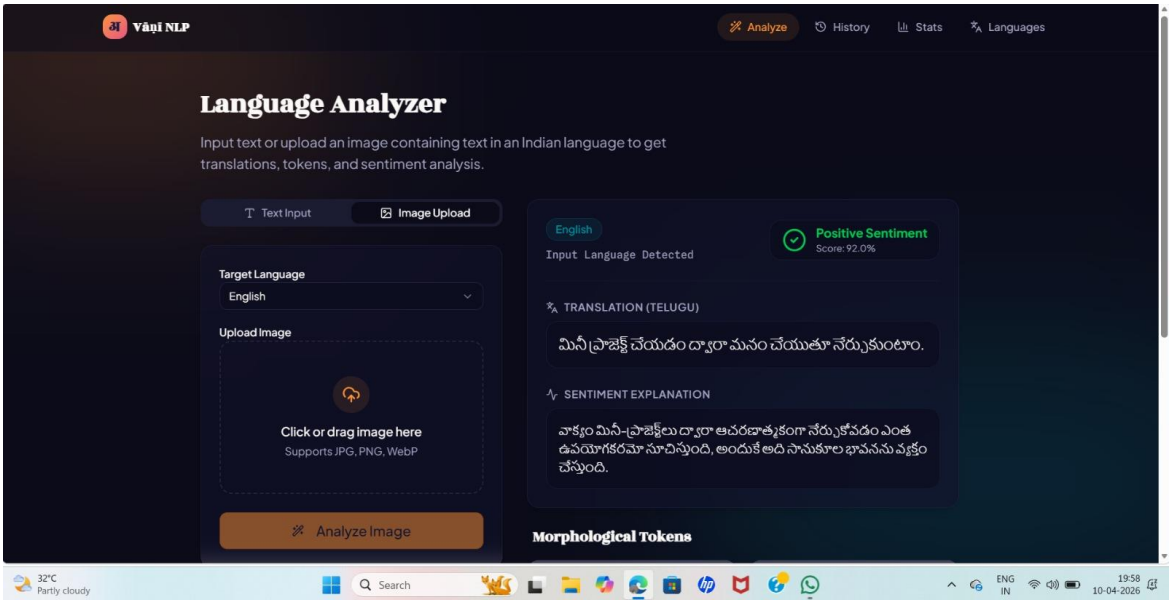


Fig 9.3. Input page of the website through Image OCR

The Image OCR input page of the system enables users to upload images containing textual content for further processing. The interface provides an option to select and upload image files, after which the system automatically extracts text using Optical Character Recognition (OCR) techniques.

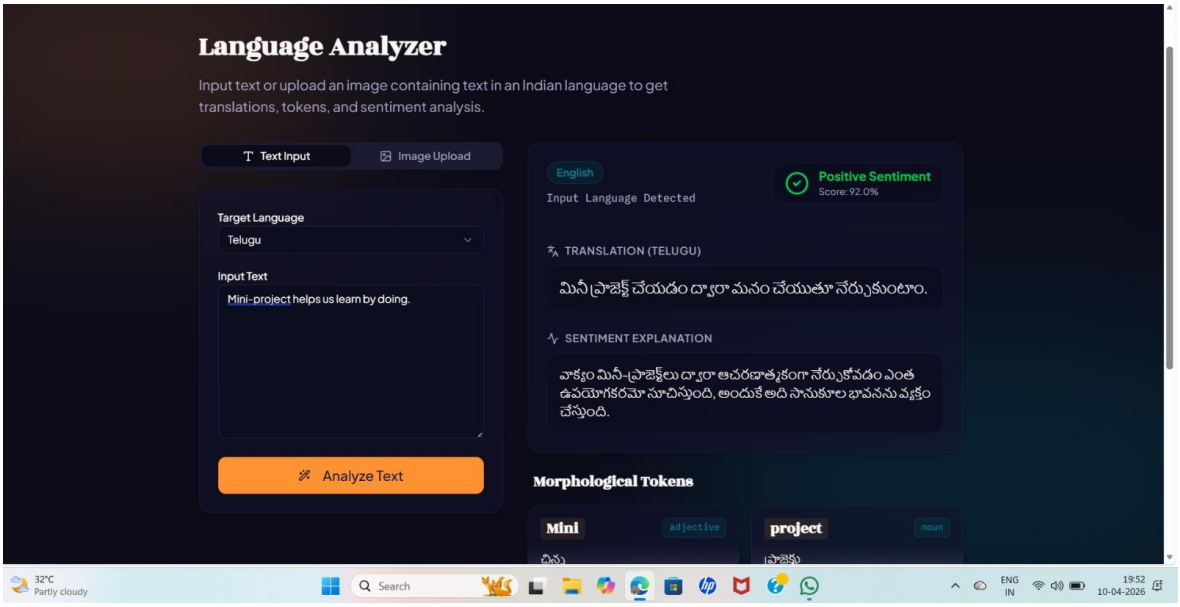


Fig 9.4. Output page of the website

Overall, the output page enhances user experience by combining multiple NLP results—translation, sentiment classification, confidence score, and original text—into a single, easy-to-interpret view, making the system both informative and user-friendly.

1. Natural language processing applications for
2. low-resource languages
3. Natural language processing applications for
4. low-resource languages
5. Natural language processing applications for
6. low-resource languages

7. Natural language processing applications for
8. low-resource languages
9. Natural language processing applications for
10. low-resource languages
11. Natural language processing applications for
12. low-resource languages
13. Natural language processing applications for
14. low-resource languages
15. Natural language processing applications for
16. low-resource language
17. Natural language processing applications for
18. low-resource languages
19. Natural language processing applications for
20. low-resource

REFERENCES:

[1]. Sabane, M., Ranade, A., Litake, O., Patil, P., Joshi, R., & Kadam, D. (2023). Enhancing Low Resource NER using Assisting Language and Transfer Learning. *2023 2nd International Conference on Applied Artificial Intelligence and Computing (ICAAIC)*, 1666-1671.

[2]. Kumar, S., Jyothi, P., & Bhattacharyya, P. (2024). Part-of-speech Tagging for Extremely Low-resource Indian Languages. *Annual Meeting of the Association for Computational Linguistics*.

[3]. Ganesh, D.B., Yaluru, G., Nilesh, D., Gummadi, A.N., Reddy, C.V., & Gangashetty, S.V. (2025). Enhancing NLP for Low-Resource Languages using Cross-Lingual Transfer and Few-Shot Learning. *2025 5th International Conference on Soft Computing for Security Applications (ICSCSA)*, 1203-1207.

[4]. Karthikeyan, J., Sayed, M.Y., Waykar, Y., Sathya, V., R, K.P., & Suseela Grace Padma, M. (2025). Natural Language Processing for Multilingual and Low-Resource Languages. *2025 IEEE 7th International Conference on Computing, Communication and Automation (ICCCA)*, 1-7.

[5]. S, S.P., P, K., T A, S.L., & A, N.K. (2025). Real-Time Sentiment Analytics: Integrating NLP for Social Media Insight. *2025 International Conference on Automation and Computation (AUTOCOM)*, 95-100.

[6]. Singh, A., Kumari, A., Singh, N., & Kumar, K. (2025). Multilingual Text-to-Speech System for Indian Languages. *2025 IEEE 7th International Conference on Computing, Communication and Automation (ICCCA)*, 1-5.

[7]. Gaikwad, P., Doshi, M., Deoghare, S.D., & Bhattacharyya, P. (2023). Machine Translation Advancements for Low-Resource Indian Languages in WMT23: CFILT-IITB's Effort for Bridging the Gap. *Conference on Machine Translation*.

[8]. Nag, A., Samanta, B., Mukherjee, A., Ganguly, N., & Chakrabarti, S. (2022). Transfer Learning for Low-Resource Multilingual Relation Classification. *ACM Transactions on Asian and Low-Resource Language Information Processing*, *22*, 1 -24.

[9]. Dhar, R., & Das, D. (2021). Leveraging Expectation Maximization for Identifying Claims in Low Resource Indian Languages. *ICON*.

[10]. Shah, S., Guha, S., Khanuja, S., & Sitaram, S. (2020). Cross-lingual and Multilingual Spoken Term Detection for Low-Resource Indian Languages. *ArXiv, abs/2011.06226*.