International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 05 | May - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Language Translation Using Marine MT

Adithya V
CEI(AIML)
Presidency University
Bengaluru,Karnataka,India
ADITHYA.20211CEI0057@presidencyuniversity.in

Tejas Gowda V
CEI(AIML)
Presidency University
Bengaluru,Karnataka,India
TEJAS.20211CEI0051@presidencyuniversity.in

Poornima Selvaraj
Senior Associate Proffesor
Presidency University
Bengaluru,Karnataka,India
poornima.s@presidencyuniversity.in

***Abstract*—In today's global landscape, effective multilingual communication is essential. Traditional machine translation methods, including rule-based and statistical approaches, have been surpassed by neural machine translation (NMT), particularly with the adoption of transformer-based models. Hugging Face, a widely-used open- source platform, offers pre-trained NMT models such as MarianMT, M2M-100, and No Language Left Behind (NLLB), enabling efficient and scalable translation across numerous language pairs.**

This paper explores the implementation of Hugging Face's translation models in real-time multilingual applications, including live chat, conferencing, and dynamic content translation. We evaluate the platform's ease of deployment, model versatility, and performance in latency-sensitive environments. Additionally, challenges such as low-resource language support, context preservation, and real-time processing constraints are examined.

Our findings highlight Hugging Face's potential in accelerating research and development in NMT, offering practical solutions for seamless cross-language communication in real-world systems.

Keywords—Language translation, Hugging Face, Python, styling, Microsoft, Textmining

I. INTRODUCTION

As global connectivity continues to expand, the need for language translation systems that are fast, accurate, and scalable has become increasingly vital. From international collaboration and real-time customer service to cross-border education, the ability to process and generate multilingual content plays a key role in breaking down communication barriers. While early translation systems relied on rule-based methods and statistical machine translation (SMT), these approaches have largely been surpassed by the capabilities of modern neural machine translation (NMT) techniques.

The advent of transformer-based models—such as BERT, GPT, and especially the Transformer architecture introduced by Vaswani et al. (2017)—has revolutionized NMT by enabling more fluent, context-aware, and generalizable translations across a wide range of language pairs. In this landscape, Hugging Face has emerged as a leading platform, offering open-source tools and pre-trained models that make advanced language technologies more accessible to both researchers and practitioners.

Through the Hugging Face Transformers library, users can leverage powerful models like MarianMT, M2M-100, and No Language Left Behind (NLLB), which collectively support translation across dozens to hundreds of languages. These models can be used directly or fine-tuned for specific domains, enabling diverse applications such as document translation, multilingual chat systems, and speech translation when integrated with external speech processing components.

Hugging Face distinguishes itself with its intuitive interface, a vast collection of community-contributed models in the Model Hub, and seamless integration with the transformers and datasets libraries. This ecosystem empowers users to quickly prototype and deploy robust translation systems without the need for extensive computational infrastructure or deep expertise in model training.

This paper investigates the application of Hugging Face models in the development and evaluation of real-time multilingual translation systems. It explores practical use cases, examines the current challenges in the field, and highlights opportunities for innovation—particularly in areas like low-resource language translation, context-sensitive modeling, and speech-to-text integration.

II. LITERATURE REVIEW

A. *Evolution of Machine Translation*

Machine translation has undergone major transformations over the years. Early systems utilized **rule-based** and **statistical methods**, such as Phrase-Based Statistical Machine Translation (PB-SMT), which were limited in handling contextual nuances and long-range dependencies. Although foundational, these approaches struggled with fluency and scalability across diverse languages.

B. *Hugging Face and the Democratization of NMT*

Hugging Face's transformers library has been instrumental in making advanced NMT models widely accessible. As highlighted by Wolf *et al.* [2], the platform simplifies the deployment of large-scale models and supports rapid experimentation. It offers plug-and-play access to a wide array of pre-trained translation models such as:

- **MarianMT** [3] – Fast and efficient, ideal for low-latency environments.
- **M2M-100** – Developed by Meta AI, supports direct translation across 100+ languages without pivoting through English.

- **No Language Left Behind (NLLB)** [4] – Designed for high-quality translation of low-resource languages, supporting 200+ languages.

C. Addressing Multilingual and Low-Resource Challenges

Multilingual models like M2M-100 and NLLB aim to reduce the imbalance in translation quality across languages. The NLLB project, in particular, emphasizes **low-resource language inclusion** by using large-scale pretraining and fine-tuning strategies to bridge quality gaps. These models are vital for improving global accessibility and digital inclusivity.

III. Methodology

This study employs pre-trained neural machine translation models available through the Hugging Face transformers library to construct a real-time multilingual translation pipeline. Models such as MarianMT, M2M-100, and No Language Left Behind (NLLB) are selected based on their extensive language support and translation performance. The system is implemented using Python, with Hugging Face's APIs facilitating streamlined integration and efficient inference via the pipeline() interface.

For evaluation, parallel corpora from the OPUS project and the FLORES-101 benchmark are used to assess translation quality, processing speed, and contextual fluency across a diverse range of language pairs. Standard metrics, including BLEU and METEOR, are calculated to quantify translation accuracy. To emulate real-time applications, latency and throughput are measured under both batch and streaming conditions. The pipeline is deployed using a lightweight Flask-based API, enabling live testing in scenarios such as chat-based multilingual communication. A comparative analysis is conducted to highlight each model's strengths, limitations, and overall effectiveness in real-time translation tasks.

A. Model Selection

Three pre-trained transformer-based models were selected based on their multilingual capabilities:

- **MarianMT** (Junczys-Dowmunt et al. [3]) for efficiency in translation tasks.
- **M2M-100** for direct translation between 100+ languages without pivoting through English.
- **No Language Left Behind (NLLB)** for its comprehensive support of 200+ languages, including low-resource ones.

These models are accessed via Hugging Face's Model Hub.

B. Dataset Preparation

- Open parallel corpora from **OPUS** and **FLORES-101** are used for evaluation. Languages were selected across high-, mid-, and low-resource categories. Text was preprocessed (tokenization and normalization) using Hugging Face's datasets library.
- Translation converts a sequence of text from one language to another. It is one of several tasks you can formulate as a sequence-to-sequence problem, a powerful framework for returning some output from an input, like translation or summarization. Translation systems are commonly used for translation between different language texts, but it can also be used for speech or some combination in between like text-to-speech or speech-to-text..
- Use a zero before decimal points: "0.25", not ".25". Use "cm3", not "cc".

C. System Implementations

The translation system is implemented in Python using Hugging Face's transformers and datasets. The pipeline() API simplifies model loading and inference. A lightweight web interface is built with Flask to simulate real-time use cases.

Example Code:

```
from transformers import pipeline


# Load a translation pipeline
translator = pipeline("translation", model="Helsinki-NLP/opus-mt-en-fr")


# Translate a sample sentence
result = translator("Hello, how are you?", max_length=50)
print(result[0]['translation_text'])
```

System implementation in language translation entails the development of a software pipeline capable of converting text or speech from one language to another. This process typically consists of multiple components, including an input module (handling either raw text or transcribed speech), a core translation engine—commonly based on neural network architectures such as Transformer models—and an output module for generating the translated text or synthesized speech. To enhance translation quality and overall system performance, pre-processing steps (e.g., tokenization, normalization) and post-processing techniques (e.g., detokenization, punctuation restoration) are often integrated. These additional layers contribute to improved linguistic accuracy, contextual coherence, and user experience across various translation scenarios. ."

D. Real-Time Deployment

- BLEU and METEOR scores for linguistic accuracy.
- Latency and throughput for performance in real-time settings.

- Pre-trained models such as MarianMT, M2M-100, or NLLB are selected based on language coverage and performance.
- The models are loaded using Hugging Face's `pipeline()` or `AutoModel` and `AutoTokenizer` classes.
- Supports both text input (via text boxes or HTTP POST) and potential speech input (integrated with ASR modules like Whisper).
- Inference is performed in real time, with translation returned within milliseconds to seconds, depending on model size.
- The translated text is returned to the user via the frontend or API response.
- Optionally integrated with TTS (text-to-speech) for spoken output in real-time applications.
- The system can be containerized using Docker and deployed on cloud platforms (e.g., AWS, GCP).
- Load balancing and asynchronous task queues (e.g., Celery + Redis) may be added for handling multiple concurrent requests.
- Do not confuse "imply" and "infer".
- A simple frontend (using HTML/CSS/JS or frameworks like React) may be developed for live translation demos, chatbots, or multilingual interfaces.

IV. EVALUTION

After the text edit has been completed, the paper is ready for the template. Duplicate the template file by using the Save As command, and use the naming convention prescribed by your conference for the name of your paper. In this newly created file, highlight all of the contents and import your prepared text file. You are now ready to style your paper; use the scroll down window on the left of the MS Word Formatting toolbar.

A. Technical Evalution

To assess the performance of the real-time language translation system, both quantitative metrics and practical indicators were used. The evaluation focused on translation accuracy, processing speed, and overall system reliability under real-time conditions.

A. Translation Accuracy

Translation quality was evaluated using two widely accepted metrics:

- **BLEU (Bilingual Evaluation Understudy):** Measures n-gram overlap between machine-generated and reference translations. Higher BLEU scores indicate closer alignment with human reference output.
- **METEOR (Metric for Evaluation of Translation with Explicit ORdering):** Considers synonymy, stemming, and word order, often correlating better with human judgment than BLEU.

B. Latency Measurement

Real-time translation latency was measured as the time (in seconds) from receiving input text to delivering the translated output. Tests were conducted on a standard GPU setup.

TABLE1 LATENCY MEASUREMENT

Model	Average Latency (sec)	Max Latency (sec)
MarianMT	0.45	0.75
M2M-100	0.72	1.05
NLLB	1.10	1.50

Batch mode inference.

C. Error Analysis

Error rate was estimated by manually evaluating 100 randomly selected translations per model across various language pairs. Three categories were considered:

- **Mistranslations** (incorrect meaning)
- **Omissions** (missing content)
- **Structural Errors** (grammatical or word order issues)

TABLE2 ERROR ANALYSIS

Model	Mistranslation (%)	Omission (%)	Total Error Rate (%)
MarianMT	8.2%	5.6%	13.8%
M2M-100	6.5%	4.1%	10.6%
NLLB	4.3%	3.7%	8.0%

NLLB consistently demonstrated higher semantic accuracy

B. Human Evalution

a) Evaluation Methodology : A group of 20 bilingual participants was recruited, fluent in at least two of the evaluated language pairs (e.g., English–French, English–Hindi, English–Swahili). Each participant was asked to interact with the deployed real-time translation system using live chat-based prompts.

*After each session, participants completed a short survey using a **5-point Likert scale** (1 = strongly disagree, 5 = strongly agree), rating the system across three dimensions:*

***1Fluency** – The naturalness and grammatical correctness of the translated text.*

2Delay *– The responsiveness of the system during real-time interactions.*

3Helpfulness *– The extent to which the translation supported successful communication.*

b) Results Summary

TABLE3 RESULT SUMMARY

Evaluation Dimension	Average Score (out of 5)
Fluency	4.2
Delay (Responsiveness)	3.9
Helpfulness	4.4

Summary

Participants generally reported that translations were fluent and contextually appropriate, with minor delays observed in longer inputs or low-resource languages. NLLB was particularly well-received in multilingual conversations due to its broader language support

c) Observations and Feedback: – Users appreciated the ***clarity and structure*** *of translations, especially in casual and conversational text.*

– ***Delay tolerance*** *varied: minor delays (under 1 second) were acceptable for most users, but noticeable pauses impacted perceived real-timeness in some cases.*

– *Suggestions included the addition of* ***back-translation*** *to verify accuracy and* ***context memory*** *for handling multi-turn conversations.*

**CHALLENGES AND INNOVATION*

1) Challenges

Despite the rapid advancements in neural machine translation (NMT) and real-time deployment platforms like Hugging Face, several challenges persist when building scalable and accurate multilingual translation systems for real-world use. These challenges span computational, linguistic, and domain-specific dimensions.

.

A. Latency vs. Accuracy Trade-Off

Real-time translation systems must balance low latency with high translation accuracy. Larger models, such as NLLB-200, offer superior quality but often introduce processing delays that hinder real-time responsiveness. Optimizing inference speed through model quantization, batching, or GPU acceleration can reduce latency, but often at the cost of translation quality, particularly in low-resource languages or complex sentence structures.

B.Multi-Speaker Tracking and Diarization

In spoken multilingual conversations, accurately identifying and segmenting multiple speakers is essential for maintaining dialogue coherence. Without robust speaker diarization, translation outputs may blend or misattribute speech content, especially in overlapping or fast-paced interactions. Integrating automatic speech recognition (ASR) systems with speaker tracking modules remains a challenging yet critical task.

C. Code-Switching and Dialects:

Real-world conversations frequently involve code-switching (switching between languages mid-sentence) and the use of regional dialects. Many pre-trained models struggle with these phenomena due to limited representation in training data. This leads to degraded fluency, semantic drift, or outright mistranslation, particularly in informal or colloquial settings.

CONCLUTION

This study validates the feasibility and effectiveness of utilizing Hugging Face's pre-trained transformer models for real-time multilingual translation. By employing advanced architectures such as MarianMT, M2M-100, and NLLB, the research demonstrates that high-quality, scalable translation systems can be implemented with relatively low infrastructure overhead. Comprehensive evaluation using standard metrics—BLEU, METEOR, and latency—combined with subjective feedback through Likert-scale surveys, offers valuable insights into both system performance and user experience.

However, several challenges must be addressed to enhance real-world applicability. Key issues include managing the trade-off between translation accuracy and response time, supporting code-switching and dialectal variations, ensuring effective speaker diarization in multi-speaker scenarios, and adapting models for specialized domains such as healthcare or law.

Future work will benefit from focusing on optimizing model efficiency, incorporating speech-to-text pipelines for end-to-end communication, and enhancing support for low- resource and underrepresented languages. As neural machine translation continues to advance, open-source frameworks like Hugging Face are expected to play a pivotal role in fostering inclusive, efficient, and accessible multilingual communication technologies.

REFERENCES

[1] 1. T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, et al., "Huggingface's transformers: State-of-the-art natural language processing", CoRR, vol. abs/1910.03771, 2019.

[2] 2. J. Gala, P. A. Chitale, A. K. Raghavan, V. Gumma, S. Doddapaneni, A. K. M, et al., "Indictrans2: Towards high-quality and accessible machine translation models for all 22 scheduled indian languages", Transactions on Machine Learning Research, 2023.

[3] 3. G. Arora, "iNLTK: Natural language toolkit for indic languages", Proceedings of Second Workshop for NLP Open Source Software (NLP-OSS), pp. 66-71, Nov. 2020.

[4] 4. S. Jagadiswarananda, Bhagavad Gita, Kolkata:Sri Ramakrishna Ashrama, 1961.

[5] 5. A. B. S. Prabhupada, "Bhagavad Gita As It Is" in bengali translation, The Bhaktivedanta Book Trust, 1972.

[6] 6. D. Halder, Bhagavad Gita In Bengali - With Original Verse And Pure Bengali Translation classic ed. Patita Paban, 2020.

[7] 7. K. Papineni, S. Roukos, T. Ward and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation", Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 311-318, 2002.

[8] 8. G. Doddington, "Automatic evaluation of machine translation quality using n-gram co-occurrence statistics", Proceedings of the Second International Conference on Human Language Technology Research, pp. 138-145, 2002.

[9] 9. K. Chen and C.-C. J. Kuo, "Lepor: A log-entropy model for automatic evaluation of machine translation", Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, pp. 1042-1051, 2011.

[10] 10. V. Mishra and R. B. Mishra, "Study of example based english to sanskrit machine translation", Polibits, no. 37, pp. 43-54, 2008.

[11] 11. V. K. Gupta, N. Tapaswi and S. Jain, "Knowledge representation of grammatical constructs of sanskrit language using rule based sanskrit language to english language machine translation", 2013 International Conference on Advances in Technology and Engineering (ICATE), pp. 1-5, 2013.

[12] 12. V. Mishra and R. Mishra, "Approach of english to sanskrit machine translation based on case-based reasoning artificial neural networks and translation rules", International Journal of Knowledge Engineering and Soft Data Paradigms, vol. 2, no. 4, pp. 328-348, 2010.

[13] 13. R. Punia, A. Sharma, S. Pruthi and M. Jain, "Improving neural machine translation for sanskrit-english", Proceedings of the 17th International Conference on Natural Language Processing (ICON), pp. 234-238, 2020.

[14] 14. R. Pandey, A. K. Ojha and G. N. Jha, "Demo of sanskrit-hindi smt system", arXiv preprint, 2018.

[15] 15. N. Koul and S. S. Manvi, "A proposed model for neural machine translation of sanskrit into english", International Journal of Information Technology, vol. 13, no. 1, pp. 375-381, 2021.

[16] 16. A. Kulkarni, "A deterministic dependency parser with dynamic programming for sanskrit", Proceedings of the second international conference on dependency linguistics (DepLing 2013), pp. 157-166, 2013.

[17] 17. A. Shukla, C. Bansal, S. Badhe, M. Ranjan and R. Chandra, "An evaluation of google translate for sanskrit to english translation via sentiment and semantic analysis", Natural Language Processing Journal, vol. 4, pp. 100025, 2023.

[18] 18. C. Wilkins, The Bhagavat Geeta or Dialogue of Krishna and Arjoon in Eighteen Lectures with Notes. C. Nourse, pp. 1785.

19. V. Vyasa, The Mahabharata, Dreamland Publications, vol. 103, 1998.