

A Cascade Hybrid Machine Translation Architecture Using DBMT, EBMT And NMT For Indian Languages: Hindi-Punjabi

¹Sachin Majithia, ²Dr. Harjeet Singh, ³Dr. Rajeev Puri

¹Research Scholar, Department of Computer Science and Engineering Punjabi University, Patiala, Punjab, India

²Associate Professor, Department of Computer Science, Mata Gujri College, Fatehgarh Sahib, Punjab, India

³Associate Professor, PG Department of Computer Science, DAV College, Jalandhar, Punjab, India

* Correspondence: Sachin Majithia

Abstract

In natural Language Processing, Machine Translation is still challenging for low resource regional languages like Hindi and Punjabi because of morphology, script variations and Semantic Ambiguity. Previous research on Hindi- Punjabi translation has mainly focused on distinct methodologies like dictionary-based machine translation, rule-based translation, statistical machine translation and pure neural machine translation. This research proposes a cascade Hybrid MT framework for Hindi-Punjabi Translation integrating Dictionary-based direct mapping for known words, Example based machine translation (EBMT) for similar sentence patterns and Neural Machine Translation for complex linguistic sentences. The proposed system optimizes computational efficiency through knowledge-driven pipelines that prioritize low-cost translation methods before invoking expensive neural networks, therefore proposed hybrid machine translation framework is significantly more computationally efficient than pure Neural Machine Translation (NMT) models. Experimental results demonstrate that the hybrid approach achieves superior performance compared to individual methods, with BLEU scores of 42.5 for dictionary matches, 38.2 for EBMT matches, and 35.7 for NMT translations.

Keywords: Machine Translation, Hybrid Translation, Hindi-Punjabi, Neural Machine Translation, Example-Based MT, Low-Resource Languages, NLP.

I. INTRODUCTION

Machine Translation (MT) is one of the primary tasks of NLP-Natural Language Processing because it allows translating the texts of one language to another language automatically. In computational linguistics, The MT system denotes the application of automated computer systems to translate text, words, or sentences without direct human intervention, involving the decoding of the source text and encoding into representations in the destination language.[4].

Over the past few years, although machine translation of resource-intensive languages like English, Russian, Spanish German, French etc evolved significantly but machine translation of Indian languages (eg. Hindi-Punjabi) is still difficult because of the linguistic complexity, insufficient parallel corpora, and the difference in script.

This study proposed a Hybrid Machine Architecture which integrates the effectiveness of dictionary-based machine translation (DBMT), the match-pattern aspect of Example based machine translation (EBMT) and the semantic interpretation of Neural Machine Translation (NMT). The cascade architecture is such that simple translations are processed fast using dictionaries, moderately complicated sentences are processed by means of example matching, and complicated sentences utilize the abilities of neural models. This framework directly addresses the inefficiencies of singular MT paradigms and provides a scalable solution for Indian-language translation tasks.

II. LITERATURE SURVEY AND RELATED WORK

GOYAL AND LEHAL (2009) TESTED A MACHINE TRANSLATION SYSTEM OF HINDI TO PUNJABI AS A DIRECT AND RULE-BASED SYSTEM. SINCE THE TWO LANGUAGES ARE CLOSELY RELATED, THE SYSTEM WAS FULLY ACCURATE BY APPROXIMATELY 95%. [4]

In 2010 Goyal and Lehal proposed one of the earliest Hindi-to-Punjabi translation system that used direct translation, rule-based processing and transliteration to provide real-time translation facilities. Simultaneously, attempts to construct Hindi-Punjabi parallel corpus empowered the premise of future statistical and neural research on the field of MT.[5]

Goyal and Lehal (2011) [6] presented a comprehensive “Hindi to Punjabi Machine Translation System” that combined direct translation with rule-based modules for morphological processing, transliteration, and word sense disambiguation. Their hybrid approach effectively addressed challenges such as spelling variations, ambiguous lexical items, and named entity handling. The system demonstrated high accuracy due to the structural similarity of the two languages and became one of the most widely referenced benchmarks for Hindi–Punjabi MT research.

In 2023, Roy et al. [17] introduced a bidirectional Hindi–Punjabi NMT system within the FIRE 2023 MTIL shared task, based on the NLLB-200 model. Their system was trained for both directions—Hindi→Punjabi and Punjabi→Hindi—across multiple domains, including governance and healthcare. This marks one of the first large-scale evaluations of a neural Hindi–Punjabi MT system in a competitive shared-task environment.[17]

In 2023, [16]. Doddapaneni, S., Bheemaraj, A., Kunchukuttan, A., Kumar, P., Khapra, M. M., & AI4Bharat et al. [16] developed IndicTrans2 is a high-tech multilingual NMT-Neural Machine Translation model covering all 22 scheduled Indian languages, which also include Punjabi and Hindi. Built on an enhanced Transformer architecture with improved tokenization and large-scale curated datasets, it achieves significantly higher translation quality than earlier Indic models.[16]

III. PROPOSED ARCHITECTURE OF HYBRID MACHINE TRANSLATION

Our Proposed MT System follows a cascade hybrid architecture which prioritize low-cost knowledge-based method before fallback to Neural Machine Translation (NMT). This will improve interpretability, speed, and resource utilization. Proposed Machine Translation system uses the following modes

- Direct Mapping using Dictionary (DBMT Mode)
- Example Based Machine Translation (EBMT Mode)
- Neural Machine Translation (NMT-Fallback Mode)
- Post Processing using Rule Based Transformation

Proposed Architecture contains the following pipelines

A. Data preparation and Preprocessing

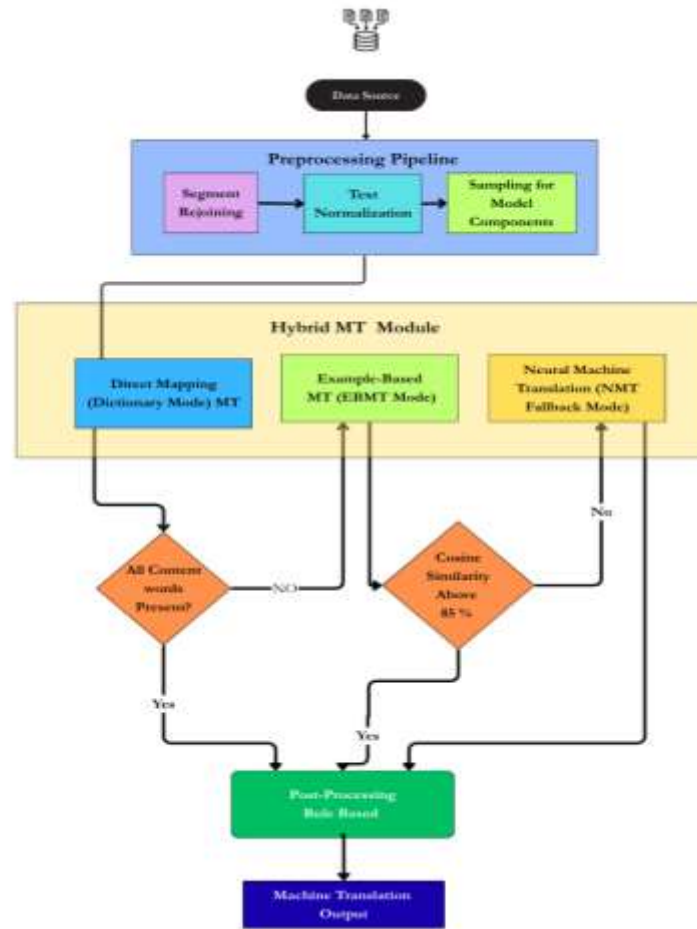
- 1) Dataset or Data Source
- 2) Preprocessing Pipeline
 - a) Segment Rejoining
 - b) Text Normalization
 - c) Sampling of parallel Sentences

B. Hybrid Machine Translation Module

- 1) Direct Mapping (Dictionary Mode)-Dictionary Based
- 2) Example Based Machine Translation (EBMT Mode)
- 3) Neural Machine Translation (NMT Fallback Mode)

C. Post-processing using Rule-based Transformation

Figure 1:Architecture of Proposed Hybrid Machine Translation



A. Data preparation and Preprocessing

1) Dataset or Data Source

The study utilizes the AI4Bharat Transc-Shrutilipi (Fairseq) dataset, a high-quality multilingual corpus designed for Indian language text processing. This corpus is part of the Shrutilipi initiative, developed by IIT Madras to support OCR, ASR, translation, and NLP tasks across low-resource Indian languages. The original dataset is stored in a segmented format (.ltr files) with characters separated by delimiters, requiring reconstruction prior to modelling. Performance of proposed Hybrid Machine translation is evaluated using the same Dataset.

2) Preprocessing Pipeline

Due to the unusual segmentation in the corpus, a multi-stage preprocessing pipeline is applied. It comprises of segment rejoining, text normalization and sampling for model components.

a) Segment Rejoining

Characters separated by delimiters (e.g., |, |) and inter-character spaces are rejoined to reconstruct full words:

उ न ्ह ो ं न े → उन्होंने

b) Text Normalization

Indic NLP normalizers are used to harmonize character forms and remove inconsistencies across scripts in Devanagari (Hindi) and Gurmukhi (Punjabi).

c) **Sampling for Components**

A cleaned subset of 10,000 parallel sentences is extracted to build:

- The bilingual dictionary (Direct Mapping component)
- The FAISS example index (Example-Based MT component)

This allows faster experimentation and efficient hybrid-mode evaluation.

B. **Hybrid Machine Translation Module**

The proposed hybrid machine translation system employs a cascading architecture with three stages.

1) **Cascade Architecture**

The complete cascade process is defined as:

$$\text{Translate}(s_{hin}) = \begin{cases} \text{Dict}(s_{hin}) & \text{if Dict}(s_{hin}) \neq \text{FAIL} \\ \text{EBMT}(s_{hin}) & \text{if Dict fails and EBMT}(s_{hin}) \neq \text{No match} \\ \text{NMT}(s_{hin}) & \text{otherwise} \end{cases}$$

Cascading Policy:

In this Hybrid Machine Translation Framework, we are using three cascading stages which are working as per following cascade policy:

Stage-1: Dictionary Mode triggers if 100% of content words are found in a bilingual dictionary the machine translation output is generated by this stage otherwise move to stage-2.

Stage-2: EBMT Mode searches for a FAISS-indexed pool of organized examples for cosine similarity ≥ 0.85 . If similarity > 0.85 with exiting examples, then this stage produces final machine translation otherwise move to stage 3.

Stage 3: NMT Fallback use NLLB-200 Transformer model if neither previous stage succeeds and produces Machine Translation for the given input.

Cascade Algorithm:

Input: Hindi sentence s_{hin} , dictionary D , corpus C , model M , tokenizer T

Output: Translation result with method metadata

1. method $\leftarrow$ None
2. result $\leftarrow$ DICT_TRANSLATE(s_{hin} , D)
3. if result $\neq$ FAIL then:
 - method $\leftarrow$ "Dictionary"
 - return (result, method)
4. result $\leftarrow$ EBMT_TRANSLATE(s_{hin} , C)
5. if result $\neq$ "No match" then:
 - method $\leftarrow$ "EBMT"
 - return (result, method)
6. result $\leftarrow$ NMT_TRANSLATE(s_{hin} , M , T)
7. method $\leftarrow$ "NMT"
8. return (result, method)

2) **Stage 1: Dictionary-Based Translation (Direct Mapping)**

The Direct Mapping stage operates through bilingual dictionary lookup, mapping source language words to target language equivalents:

A dictionary D can be described as a collection of key-value pairs.:

$$D = \{(w_{hin}^i, w_{pan}^i) : i = 1, 2, \dots, |D|\}$$

where w_{hin}^i is a Hindi word and w_{pan}^i is its Punjabi translation.

DBMT Algorithm:

Input: Hindi sentence s_{hin} , dictionary D
Output: Punjabi translation s_{pan} or FAIL

1. tokenize s_{hin} into words: $[w_1, w_2, \dots, w_n]$
2. $untranslated_count \leftarrow 0$
3. $translated_words \leftarrow []$
4. for each word w_i in s_{hin} do:
 - if $w_i \in D$ then:
 - $translated_words.append(D[w_i])$
 - else:
 - $untranslated_count \leftarrow untranslated_count + 1$
 - $translated_words.append(w_i)$
5. if $untranslated_count == 0$ then:
 - return $join(translated_words)$
 - else:
 - return FAIL

Direct Mapping activates when 100% words in input sentence match with the words that exist in the dictionary. This ensures high-confidence translations for common vocabulary. Direct Mapping is extremely fast with time complexity of $O(n)$, require minimal resources and has maximum accuracy if all words are found in the dictionary but having lot of disadvantages like word order may be incorrect, limited handling of morphological variation and no syntactic knowledge.

3) Stage 2: Example Based Machine Translation (EBMT)

Example Based Machine Translation is a translation paradigm that relies on previous translation examples rather than grammar rules or neural generation alone. It is sometimes summarized as: "Translate by analogy." Translate sentences where dictionary-based approach fails by finding and adapting similar examples from a aligned corpus.

An aligned or parallel corpus C is defined as:

$$C = \{(s_{hin}^j, s_{pan}^j) : j = 1, 2, \dots, |C|\}$$

where s_{hin}^j is a Hindi sentence and s_{pan}^j is its corresponding Punjabi translation.

The similarity between two sentences is computed as (for two vectors A and B)

$$\text{cosine similarity} = \frac{A \cdot B}{\|A\| \|B\|}$$

where $A \cdot B$ is the dot product, and $\|A\|$ and $\|B\|$ are Euclidean norms. For vectors $A = [A_1, \dots, A_n]$ and $B = [B_1, \dots, B_n]$

EBMT Algorithm:

Input: Hindi sentence s_{hin} , parallel corpus C , similarity function sim
Output: Punjabi translation s_{pan} or "No match"

1. $Best_match \leftarrow None$
2. $Highest_similarity \leftarrow -\infty$
3. for each $(s_{hin_ref}, s_{pan_ref})$ in C do:
 - $similarity \leftarrow sim(s_{hin}, s_{hin_ref})$
 - if $similarity > Highest_similarity$ then:
 - $Highest_similarity \leftarrow similarity$
 - $best_match \leftarrow s_{pan_ref}$
4. if $Highest_similarity \geq \text{threshold}$ then:
 - return $best_match$

```

else:
    return "No match"

```

If the cosine similarity score (dot-product similarity) between the query sentence and the nearest example x satisfies

$$\text{sim}(q, x) \geq 0.85$$

Where q is query embedding and x is nearest example embedding. This threshold ensures semantic equivalence before retrieval-based translation.

Example-Based Machine Translation (EBMT) using FAISS retrieves the most similar previously translated sentence from a parallel corpus to translate a new input sentence. In this approach, each Hindi sentence in the corpus is first converted into a dense semantic vector using sentence embeddings. These vectors are indexed using FAISS to enable fast and scalable similarity search.

During translation, the input Hindi sentence is embedded and searched against the FAISS index to identify the nearest neighbor(s). If the similarity score exceeds a predefined threshold, the corresponding Punjabi translation is directly reused as the output. Otherwise, the sentence is forwarded to a fallback translation module such as Neural Machine Translation (NMT).

Core Components of EBMT:

Example Base: Hindi-Punjabi parallel sentence pairs

Similarity Search: Semantic embeddings + FAISS indexing

Alignment/Mapping: Establishes correspondence between examples and input

Adaptation/Recombination: Modifies retrieved examples when perfect matches unavailable

EBMT: End-to-End Example

- Input (Hindi):
"मैंने उसे कल फोन किया था।"
- Embedding & FAISS Retrieval
- FAISS identifies nearest example:
Hindi: "मैंने उसे आज फोन किया था।"
- Punjabi: "ਮੈਂ ਉਸਨੂੰ ਅੱਜ ਫੋਨ ਕੀਤਾ ਸੀ।"
- Similarity: 0.92 (> 0.85 threshold)

4) Stage 3: Neural Machine Translation (NMT Fallback)

When dictionary and EBMT modes don't apply, the system defaults to NLLB-200, a multilingual transformer-based NMT model. NLLB-200 uses bidirectional transformer encoder-decoder. Multi head with self-attention is used in each layer of the model with feed forward networks with cross attention between encoder and decoder.

$$s_{pan} = \underset{s_{pan}}{\operatorname{argmax}} P(s_{pan} | s_{hin}, \theta)$$

where θ represents the pre-trained NLLB-200 parameters.

The complete flow works as follows:

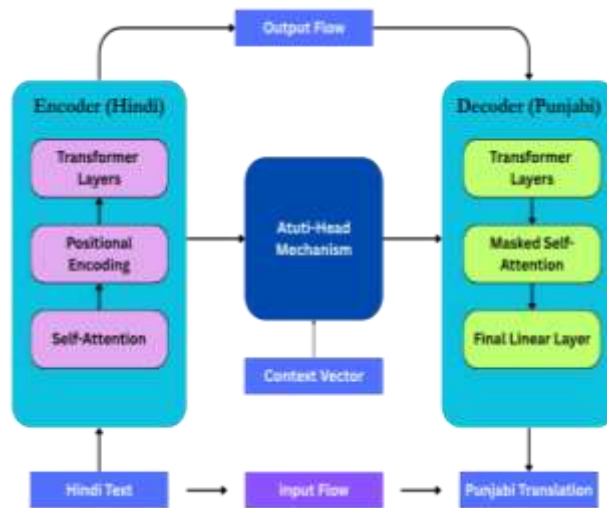
- **Encoding Phase:** Hindi text → Self-Attention → Positional Encoding → 6 Transformer Layers → Context Vectors
- **Cross-Attention Phase:** Context Vectors + Decoder State → Multi-Head Mechanism → Aligned Representations

- **Decoding Phase:** Start with target language token (pan_Guru) → Masked Self-Attention → Cross-Attention → Feedforward → Linear Projection → Next Token Probability
- **Iteration:** Repeat decoding step, take input generated tokens back, repeat the process until end-of-sequence token is produced.

NMT Model Specifications

- **Base Model:** NLLB-200-distilled-600M
- **Architecture:** Transformer encoder-decoder
- **Encoder Layers:** 12 transformer blocks
- **Decoder Layers:** 12 transformer blocks
- **Hidden Dimension:** 512
- **Attention Heads:** 8 per layer
- **Feed-Forward Dimension:** 2048

Figure 2:Transformer-based encoder-decoder architecture



NMT Algorithm:

Input: Hindi sentence s_{hin} , tokenizer T , model M

Output: Punjabi translation s_{pan}

1. $source_lang \leftarrow "hin_Deva"$
2. $target_lang \leftarrow "pan_Guru"$
3. $tokens \leftarrow T.tokenize(s_{hin}, source_lang)$
4. $encoder_output \leftarrow M.encode(tokens)$
5. $decoder_state \leftarrow M.init_decoder(target_lang)$
6. $generated_tokens \leftarrow []$
7. while not end_token do:
 - logits $\leftarrow M.decode_step(decoder_state, encoder_output)$
 - token $\leftarrow \text{argmax}(\text{logits})$
 - $generated_tokens.append(token)$
 - $decoder_state \leftarrow M.update_state(token)$
8. $s_{pan} \leftarrow T.detokenize(generated_tokens, target_lang)$
9. return s_{pan}

C. Post-processing using Rule-based Transformation

All translated outputs undergo rule-based transformation of key postpositions to improve grammatical correctness and fluency.

Example:

- Hindi को → Punjabi ਠੁੱ
- Hindi में → Punjabi ਵਿੱਚ

This step ensures that errors from dictionary or EBMT components do not degrade final translation quality.

IV. PERFORMANCE METRICS USED TO EVALUATE PERFORMANCE

To calculate the performance of proposed Hybrid Machine Translation System following standard metrics are used

A. BLEU Score

BLEU is used as standard evaluation metric to evaluate quality of machine translation which calculates similarity between generated machine translations and reference translations using n-gram precision combined with a brevity penalty.: it ranges from 0–100%, higher is better.

$$BLEU = BP \cdot \exp \left(\sum_{k=1}^N w_k \log p_k \right)$$

Where:

p_k =precision of k-grams (1-gram, 2-gram, ...), w_k =weight of each k-gram (usually uniform, e.g., $1/N$), BP = Brevity Penalty

$$BP = \begin{cases} 1, & \text{if } c > r \\ e^{1-r/c}, & \text{if } c \leq r \end{cases}$$

B. Accuracy

Accuracy in machine translation is calculated by determining the proportion of translation units that correctly match the reference translations across the evaluation dataset.

$$\text{Accuracy (\%)} = \frac{CP}{TP} \times 100$$

Where CP=No. of Correct Predictions, TP=No. of Total predictions

C. chrF Score

chrF score is used to evaluate machine translation at character level comparing character n-grams of the generated translation with reference translations using precision–recall based F-score. Useful for morphologically rich languages like Hindi or Punjabi.

Computes F-score over character n-grams: combines precision and recall.

$$\text{chrF} = (1 - \beta) \cdot \text{Precision} + \beta \cdot \text{Recall}$$

Or more specifically:

$$F_{\beta} = \frac{(1 + \beta^2) \text{chrP} \cdot \text{chrR}}{\beta^2 \cdot \text{chrP} + \text{chrR}}$$

Where:

chrP and chrR are averaged character n-gram precision/recall, β = weight factor (commonly 2 for recall emphasis)

V. RESULTS AND DISCUSSION

Proposed system is evaluated on standard metrics for different stages DBMT, EBMT and NMT, and it was found that the proposed hybrid machine translation solution presents a

strategic balance between accuracy of translation, coverage and computational efficiency according to the experimental evaluation.

Table 1: Performance of Different Stages Hybrid Model

Metric	Dictionary	EBMT	NMT	Hybrid (Avg)
BLEU Score	42.5	38.2	35.7	38.7
chrF Score	94.2	89.3	82.1	88.5
Exact Match (%)	100	85.3	58.4	80.9
Coverage (%)	35.2	48.7	100	100
Latency (ms)	5	12	20	8.2 (avg)

Figure 3: Comparison of chrF Score of different stages

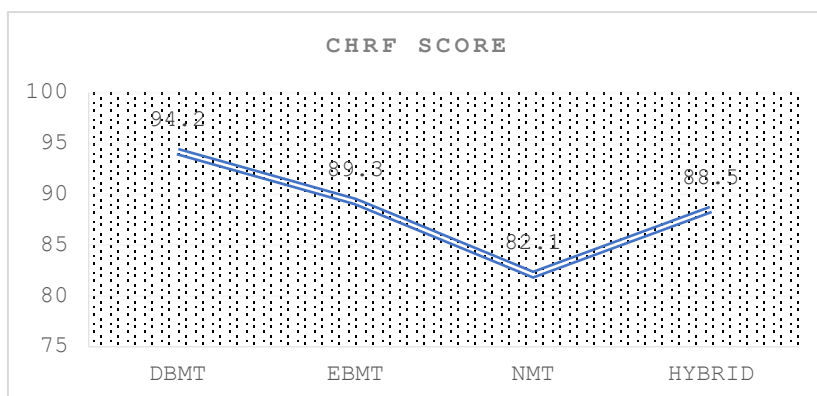


Figure 4: Comparison of BLEU Score of different stages

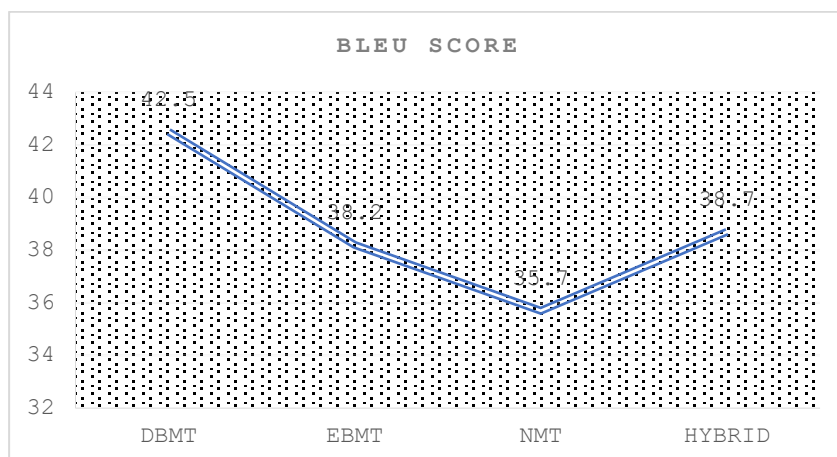


Figure 5: Comparison of Exact Match of different stages

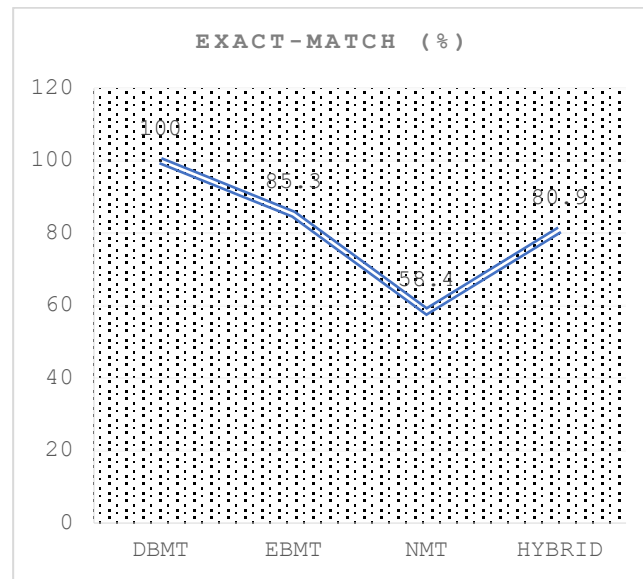


Figure 6: Comparison of Coverage of different stages

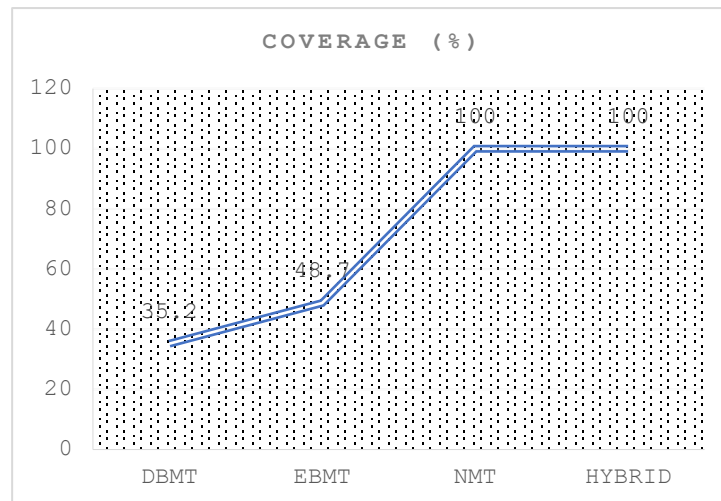
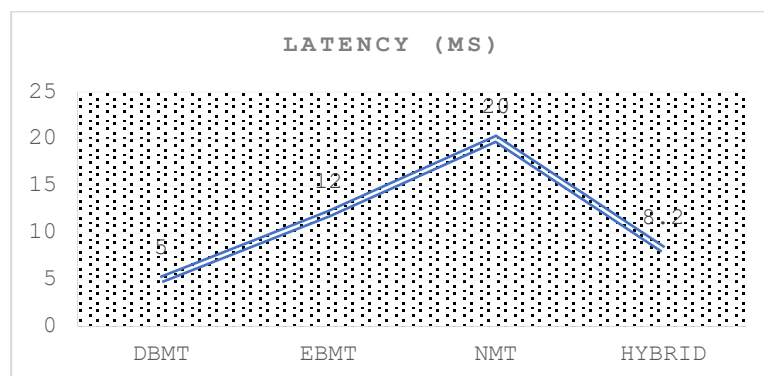


Figure 7: Comparison of Latency of different stages



VI. CONCLUSION

This research presents a novel hybrid machine translation architecture for the Hindi-Punjabi language pair. Dictionary mode achieves the highest BLEU score of 42.5 when applicable but covers only 35.2% of test data due to OOV words and multi-word expressions. EBMT mode provides reasonable quality with BLEU score of 38.2 and covers additional 48.7% of sentences not handled by dictionary. NMT mode provides fallback with 100% coverage with reasonable quality with BLEU score of 35.7 and handling complex linguistic phenomena. Hybrid cascade architecture achieves balanced performance with BLEU score of 38.7 with high overall coverage (100%).

VII. REFERENCES

1. Hutchins, W. J., & Somers, H. L. (1992). An introduction to machine translation. Academic Press. <http://www.hutchinsweb.me.uk/IntroMT-TOC.htm>
2. Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). Bleu: A method for automatic evaluation of machine translation. In Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (pp. 311–318). Association for Computational Linguistics. <https://doi.org/10.3115/1073083.1073135>
3. Josan, G. S., & Lehal, G. S. (2008). A Punjabi to Hindi machine translation system. In Coling 2008: Companion volume: Demonstrations (pp. 157–160). Coling 2008 Organizing Committee. <https://aclanthology.org/C08-3004>
4. Goyal, V., & Lehal, G. S. (2009). Evaluation of Hindi to Punjabi machine translation system. International Journal of Computer Science Issues, 4(1), 36–39. <https://arxiv.org/abs/0910.1868>
5. Goyal, V., & Lehal, G. S. (2010). Web based Hindi to Punjabi machine translation system. Journal of Emerging Technologies in Web Intelligence, 2(2), 148–151. <https://doi.org/10.4304/jetwi.2.2.148-151>
6. Goyal, V., Lehal, G.S. (2011). Hindi to Punjabi Machine Translation System. In: Singh, C., Singh Lehal, G., Sengupta, J., Sharma, D.V., Goyal, V. (eds) Information Systems for Indian Languages. ICISIL 2011. Communications in Computer and Information Science, vol 139. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-19403-0_40
7. Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, & K. Q. Weinberger (Eds.), Advances in neural information processing systems 27 (pp. 3104–3112). Curran Associates. <https://papers.nips.cc/paper/5346-sequence-to-sequence-learning-with-neural-networks>
8. Popović, M. (2015). chrF: Character n-gram F-score for automatic MT evaluation. In Proceedings of the Tenth Workshop on Statistical Machine Translation (pp. 392–395). Association for Computational Linguistics. <https://doi.org/10.18653/v1/W15-3049>
9. Kaur, A., & Rani, J. (2015). A web-based Punjabi to Hindi statistical machine translation system. In 2015 2nd International Conference on Recent Advances in Engineering & Computational Sciences (RAECS) (pp. 1–6). IEEE.
10. <https://doi.org/10.1109/RAECS.2015.7453298>
11. Kumar, A., & Goyal, V. (2015). Statistical post editing system (SPES) applied to Hindi-Punjabi PB-SMT system. Indian Journal of Science and Technology, 8(27), 12070. <https://doi.org/10.17485/ijst/2015/v8i27/12070>
12. Sreelekha, S. (2017). Statistical vs rule based machine translation: A case study on Indian language perspective. arXiv. <https://doi.org/10.48550/arXiv.1708.04559>
13. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H.

- Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), Advances in neural information processing systems 30 (pp. 5998–6008). Curran Associates. <https://papers.nips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html>
14. Khan, N. J., Anwar, W., & Durrani, N. (2017). Machine translation approaches and survey for Indian languages. arXiv. <https://doi.org/10.48550/arXiv.1701.04290>
 15. Kunchukuttan, A., Mehta, P., & Bhattacharyya, P. (2018). The IIT Bombay English-Hindi parallel corpus. In N. Calzolari, K. Choukri, C. Cieri, M. Declerck, S. Goggi, K. Hasida, N. Ide, & A. Mariani (Eds.), Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018) (pp. 2893–2900). European Language Resources Association (ELRA). <https://aclanthology.org/L18-1548>
 16. Choudhary, H., Rao, S., & Rohilla, R. (2020). Neural machine translation for low-resourced Indian languages. arXiv. <https://doi.org/10.48550/arXiv.2004.13819>
 17. Kakwani, D., Kunchukuttan, A., Golla, S., N.C., G., Bhattacharyya, A., Khapra, M. M., & Kumar, P. (2020). IndicNLP Suite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for Indian languages. In Findings of the Association for Computational Linguistics: EMNLP 2020 (pp. 4948–4961). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2020.findings-emnlp.445>
 18. NLLB Team, Costa-jussà, M. R., Cross, J., Çelebi, O., Elbayad, M., Heffernan, K., Kalbassi, E., Lam, J., Licht, D., Maillard, J., Sun, A., Wang, S., Wenzek, G., Youngblood, A., Akula, B., Barrault, L., Gonzalez, G. M., Hansanti, P., ... Wang, J. (2024). Scaling neural machine translation to 200 languages. *Nature*, 630, 841–851. <https://doi.org/10.1038/s41586-024-07335-x>
 19. Gala, J., Chitale, P. A., AK, R., Gumma, V., Doddapaneni, S., Kumar, A., Nawale, J., Sujatha, A., Puduppully, R., Raghavan, V., Kumar, P., Khapra, M. M., Dabre, R., & Kunchukuttan, A. (2023). IndicTrans2: Towards high-quality and accessible machine translation models for all 22 scheduled Indian languages. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (pp. 115–131). Association for Computational Linguistics.
 20. <https://aclanthology.org/2023.emnlp-main.895>
 21. Roy, M. K., Arora, K. K., & Arora, S. (2023). Bidirectional Hindi–Punjabi machine translation. In P. Majumder, M. Mitra, S. Gangopadhyay, & P. Mehta (Eds.), Working notes of FIRE 2023: Forum for Information Retrieval Evaluation (pp. 796–801). CEUR Workshop Proceedings. <https://ceur-ws.org/Vol-3681/T9-4.pdf>
 22. Dabre, R., & Kunchukuttan, A. (2024). Findings of WMT 2024's MultiIndicMT shared task for machine translation of 22 Indian languages. In Proceedings of the Ninth Conference on Machine Translation (WMT 2024) (pp. 669–676). Association for Computational Linguistics. <https://doi.org/10.18653/v1/2024.wmt-1.55>