CS779 Competition: Machine Translation System for India

Kishan Kumar Mishra
251110044
kishankm25@iitk.ac.in
Indian Institute of Technology Kanpur (IIT Kanpur)

Abstract

We present a three-layer Transformer-based neural machine translation system for English to Hindi and English to Bengali. The encoder uses 300-dimensional GloVe embeddings for English, while the decoders use FastText embeddings for Hindi and Bengali. The systems were evaluated using BLEU scores on a held-out validation set, with decoding performed through both greedy and beam search strategies. The combined BLEU score was 0.165 on the validation set and 0.172 on the test set.

1 Competition Result

Codalab Username: k_251110044
Final Leaderboard Rank on the Test Set: 24
chrF++ Score corresponding to the Final Rank: 0.381
ROUGE Score corresponding to the Final Rank: 0.449
BLEU Score corresponding to the Final Rank: 0.172
Total Number of Submissions in the Training Phase: 28
Total Number of Submissions in the Testing Phase: 5

Table 1: Number of Submissions Made Each Week During the Training Phase

Week	Number of Submissions
1st Week	3
2nd Week	8
3rd Week	4
4th Week	13

2 Problem Description

The competition task is to build neural network based translation systems for two independent subtasks: English to Hindi and English to Bengali. Each subtask could either share a common model or use a separate, language-specific one. Any neural network architecture such as Seq2Seq, Transformer etc can be used. We are also free to use any static word embeddings. However, pre-trained models and subword tokenization libraries cannot be used directly. The goal is to train the models from scratch and evaluate their translation quality using standard metrics like BLEU.

3 Data Analysis

1. Description of Training Dataset

The datasets provided consist of parallel sentence pairs for two translation tasks: English–Bengali and English–Hindi. The English-Bengali dataset consists of **68,849** sentences whereas the English-Hindi dataset consists of **80,797** sentences.

2. Corpus Statistics and Characteristics

Table 2: Corpus statistics for English–Hindi training data (source: English, target: Hindi).

Statistic	**English Sentences**	**Hindi Sentences**
Number of sentences	80,797	80,797
Average length (tokens)	16.43	18.34
Median length (tokens)	15.0	17.0
Maximum length (tokens)	258	216
90th percentile length	28	31
95th percentile length	33	37
99th percentile length	45	50

Table 3: Corpus statistics for English–Bengali training data (source: English, target: Bengali).

Statistic	**English Sentences**	**Bengali Sentences**
Number of sentences	68,849	68,849
Average length (tokens)	16.11	14.00
Median length (tokens)	15.0	13.0
Maximum length (tokens)	87	77
90th percentile length	27	23
95th percentile length	31	27
99th percentile length	41	35

3. Comparative Observations and Insights

A comparison of the two corpora reveals several interesting patterns:

- The **Hindi corpus** is larger ($\approx$ 80K pairs) than the **Bengali corpus** ($\approx$ 68K pairs), providing more data for model generalization.
- Due to morphological richness, Bengali sentences tend to be shorter but more compact in meaning.
- Both corpora contain a small proportion of extremely long sequences(concatenated sentences or translation noise), which require truncation.
- Both Bengali and Hindi sentences occasionally contained English words, which were removed during data cleaning.
- The Hindi dataset's larger size and closer syntactic alignment with English contributed to its higher BLEU scores and more stable convergence during model training.

4 Model Description

1. **Model Evolution:** The initial set of experiments focused on GRU-based encoder-decoder architectures. In translation tasks, understanding both preceding and succeeding words provides stronger semantic alignment. Hence our first model employed a **bidirectional GRU encoder**.

   Subsequently **attention mechanism** (Bahdanau-style) was integrated into the GRU-based architecture, allowing the decoder to dynamically focus on the most relevant parts of the source sentence during each decoding step. This attention-augmented GRU model provided the most significant improvement in BLEU scores among all RNN-based systems. However, subsequent experiments with stacked recurrent layers did not yield additional performance gains, likely due to the relatively high learning rate (1×10^{-3}), which caused unstable training and limited the benefits of increased model depth.

   After achieving a BLEU score of around 10 on the development set with the attention-based GRU, focus shifted to Transformer architectures. Several variations were explored by adjusting the embedding size (100, 200, and 300), number of attention heads, and number of encoder-decoder layers. Increasing the embedding size and number of layers improved performance up to three layers, after which no significant improvement was observed. When deeper Transformers were used, reducing the learning rate improved stability and convergence, confirming the sensitivity of the model to learning rate selection.

   Both randomly initialized and pretrained word embeddings were tested with the Transformer models. Pretrained embeddings consistently achieved higher BLEU scores and faster convergence, requiring fewer epochs to reach optimal performance. However, with sufficient training, models initialized with random embeddings achieved comparable results. Among embedding dimensions, larger vectors (300-dimensional) performed better than smaller ones (100 or 200), providing richer semantic representations and contributing to more accurate translations.

2. **Inspirations from Existing Models:** The model architecture and design choices were influenced by prior works in neural machine translation [1, 2, 3].

3. **Final Model Description:** The final and best-performing model was a **Transformer-based architecture** designed for efficient sequence-to-sequence translation. The key configuration details are as follows:

   - **Architecture:** Encoder-decoder Transformer with multi-head self-attention mechanism.
   - **Embedding Dimension:** 300
   - **Number of Attention Heads:** 8
   - **Encoder and Decoder Layers:** 3 each
   - **Learning Rate Schedule:** Warm-up schedule with a peak learning rate of 3.5×10^{-4} for stable convergence.
   - **Pretrained Embeddings:** GloVe embeddings for English and FastText embeddings for Hindi and Bengali, enabling faster convergence and improved BLEU scores.
   - **Decoding Strategy:** Beam search decoding with a beam width of 5 for generating more fluent and contextually accurate translations.
   - **Training Duration:** 60 epochs, with early stopping based on validation performance.
   - **Maximum Sentence Length:** 30 tokens for Hindi and 20 tokens for Bengali, selected empirically to balance model capacity and training stability.

   This configuration achieved the highest BLEU score on both development and test sets, demonstrating stable convergence and strong generalization across language pairs.

4. **Model Objective (Loss) Function:** The loss function used was the standard cross-entropy loss between predicted and true token distributions:

$$\mathcal{L} = -\sum_{t=1}^{T} \log P(y_t | y_{<t}, x)$$

where y_t is the target token at time step t, and x represents the input sequence.

5. **Inference and Decoding Strategy:** During inference, **greedy decoding** was initially used for simplicity and quick evaluations. Later, **beam search decoding** with a beam width of 5 was adopted, which led to more fluent and contextually accurate translations. The switch from greedy to beam search was motivated by the consistent improvements in BLEU scores with minimal additional inference cost.

5 Experiments

1. **Data Pre-processing:** For all three languages (Hindi, English, and Bengali), the following pre-processing steps were applied:

   - Removal of punctuations and any character not belonging to the respective language alphabet.
   - Tokenization using `nltk.word_tokenize`.
   - Conversion of all text to lowercase.

   The purpose of this pre-processing was to ensure clean and consistent textual input by eliminating noise (such as symbols and mixed-script characters) and standardizing token representation across the datasets.

2. **Training Procedure:** The models were trained using the Adam optimizer with a warm-up learning rate schedule.

   - Peak learning rate: 3.5×10^{-4}
   - Warm-up strategy: gradual increase in learning rate during initial steps, followed by decay.
   - Number of epochs: 60

3. **Hyper-parameter Details:** The following table summarizes the hyper-parameters used for different models and their selection rationale.

Table 4: Hyper-parameter settings for GRU baseline and Transformer (Hindi/Bengali) models.

Parameter	**GRU Baseline**	**Transformer (Hindi/Bengali)**
Optimizer	Adam	Adam
Learning Rate (peak)	1×10^{-3}	3.5×10^{-4}
Warm-up Schedule	No	Yes (Adaptive)
Epochs	20	60
Number of Layers	1 encoder + 1 decoder	3 encoder + 3 decoder
Hidden Size	512	512
Embedding Dimension	300	300
Batch Size	64	64

The hyper parameters were finalized through iterative experimentation and observation of validation performance.

6 Results

Table 5: Results of different models on the development dataset with evaluation metrics and ranks.

No.	Description / Configuration	Evaluation Metrics (BLEU / ROUGE / CHRF)	Remarks / Notes	Rank
1	Bidirectional GRU, hidden size=128, embedding size=128, lr 1e-3, 10 epochs, max seq len 20	0.047 / 0.278 / 0.212	Baseline GRU model	Top 5
2	Same as Model 1, 20 epochs	0.049 / 0.272 / 0.217	Longer training	Top 5
3	GloVe embeddings, hidden size 512, embedding size=100	0.056 / 0.308 / 0.252	Pretrained embeddings improved performance	Top 5
4	2 layered GRU. Vocab construction from training data only	0.056 / 0.319 / 0.256	No improvement(maye because of high lr which destabilised learning)	Top 5
5	Used sum of encoder hidden states instead of the final hidden state	0.059 / 0.330 / 0.267	Slight improvement	Top 5
6	Single layer in decoder	0.062 / 0.333 / 0.275	Simplified decoder	Top 5
7	Single layer in encoder	0.064 / 0.338 / 0.280	Improved encoder representation	Top 5
8	Used cross attention between decoder and encoder	0.065 / 0.339 / 0.285	Attention improves score	Top 5
9	Concatenated bidirectional hidden states instead of summing	0.088 / 0.359 / 0.322	Major improvement due to concatenation	Top 5
10	Frequency cut-off = 2 for Hindi	0.088 / 0.341 / 0.297	Slight drop due to vocabulary trimming	Top 5
11	Embedding size=200, lr reduced to 1e-3 from 3e-3	0.098 / 0.370 / 0.334	Improved with lower lr and larger embedding	Top 5
12	Single layered Transformer with d_model=200, embedding size=200. Only Hindi model updated.	0.111 / 0.366 / 0.329	Transformer introduced for Hindi sub task. Only Hindi model updated.	Top 5
13	Embedding size=300, d_model=300. Only Hindi model updated.	0.123 / 0.389 / 0.347	Better results with larger embedding.	Top 5
14	2 layered Transformer. Only Hindi model updated.	0.127 / 0.387 / 0.342	Slight improvement with deeper model.	Top 5
15	3 layered Transformer. Only Hindi model updated.	0.130 / 0.388 / 0.351	Best result among shallow Transformers	Top 5
16	3 layers with projection layer, lr 1e-4. Only Hindi model updated.	0.139 / 0.403 / 0.351	Lower lr improved convergence	Top 5
17	Model 16 applied to Bengali as well	0.147 / 0.414 / 0.353	Improvement due to bengali sub task optmization	Top 5

Continued on next page

Table 5 continued from previous page

No.	Description / Configuration	Evaluation Metrics (BLEU / ROUGE / CHRF)	Remarks / Notes	Rank
18	GELU + PreNorm. Only Hindi model updated.	0.141 / 0.410 / 0.363	Alternate activation and normalization. Not used in further models.	Top 5
19	3 layers, projection 768, 12 heads. Only Hindi model updated.	0.145 / 0.416 / 0.369	No gain. Not used further.	Top 5
20	FastText embeddings, 3 layers, 40 epochs.	0.147 / 0.414 / 0.370	Faster convergence within 20 epochs and only slight improvement after that	Top 5
21	Warm-up LR schedule (peak 3.5e-4), label smoothing	0.147 / 0.411 / 0.352	Faster convergence with 15 epochs and only slight improvement after that	Top 5
22	Cleaned text data by removing out of language alphabet chars	0.158 / 0.424 / 0.384	Cleaning improved all metrics	Top 5
23	IndicNLP usage	0.152 / 0.422 / 0.380	Worse performance than nltk word_tokenize. Not used further.	Top 5
24	Reduced training sentence length to 15	0.151 / 0.422 / 0.382	Not an optimal length for training	Top 5
25	Sentence length=20 and Beam search decoding	0.164 / 0.430 / 0.398	Beam search performed better than greedy decoding	Top 10
26	Tuned beam search	0.164 / 0.430 / 0.394	No improvement	Top 10
27	Increased number of epochs to 60(from 40)	0.165 / 0.433 / 0.396	Negligible improvement	10th
28	4 layered transformer	0.162 / 0.430 / 0.395	Worsened performance	10th

Table 6: Results of different models on the test dataset with evaluation metrics.

No.	Description / Configuration	Evaluation Metrics (BLEU / ROUGE / CHRF)	Remarks / Notes
1	25th submission from training phase but with greedy decoding	0.156 / 0.429 / 0.380	Baseline test model
2	Same as Test Model 1 but used beam search decoding for Hindi	0.152 / 0.428 / 0.380	Minor drop in BLEU score, an exception since beam search generally performs better than greedy decoding
3	Same as Test Model 1 but trained for 60 epochs	0.158 / 0.432 / 0.385	Improved Bengali translations
4	Same as Test Model 4 but used beam search for Bengali	0.160 / 0.437 / 0.392	Slight improvement

Continued on next page

Table 6 continued from previous page

No.	Description / Configuration	Evaluation Metrics (BLEU / ROUGE / CHRF)	Remarks / Notes
5	Increased maximum sentence length to 30 for Hindi sub task. Used beam search decoding for Hindi and Bengali	0.172 / 0.449 / 0.381	Best overall test performance; improved handling of longer Hindi sentences

The **fifth model** achieved the best overall performance, with BLEU, ROUGE, and CHRF scores of **0.172**, **0.449**, and **0.381**, respectively. This improvement can be attributed to increasing the maximum sentence length for Hindi from 20 to 30 tokens, which allowed the model to better capture long-range dependencies and syntactic structures.

7 Error Analysis

Model-wise Performance and Observations

Across experiments, the GRU-based sequence-to-sequence models demonstrated strong baseline performance and effectively captured short and medium-range dependencies. The introduction of attention mechanism further enhanced their ability to dynamically focus on relevant source tokens, leading to noticeable improvements in translation fluency and alignment. Building upon these results, Transformer-based architectures were subsequently explored to better model global context and parallelize training.

The Transformer-based models significantly improved translation quality on both the development and test sets. The self-attention mechanism enabled better handling of global dependencies, and positional encodings preserved word order effectively. Among the Transformer variants, the three-layer encoder-decoder architecture with eight attention heads and 300-dimensional pretrained embeddings achieved the best balance between performance and stability.

Additionally, **beam search decoding** consistently outperformed greedy decoding, providing an average gain of approximately 0.5–1.0 BLEU point across both Hindi and Bengali translation tasks. The use of a beam width of 5 allowed the model to explore multiple hypotheses during decoding, leading to more fluent and contextually accurate outputs with minimal additional computational overhead.

For the Hindi language subtask, increasing the maximum sentence length during training further improved translation quality, yielding an increase of about 1.2 points in the combined BLEU score on the test set. This suggests that the model benefited from exposure to longer sequences, enabling it to better capture syntactic dependencies and context beyond short sentence boundaries.

Common Errors and Possible Improvements

Despite substantial progress, several types of errors were consistently observed:

- **Literal Translations:** The model often produced word-by-word translations that failed to capture context-dependent meanings.
- **Word Repetition in Weaker Models:** Earlier GRU-based models frequently exhibited repetitive output patterns, where a single token or short phrase was repeated toward the end of the sentence. This typically occurred due to exposure bias and insufficient coverage of long-range dependencies during decoding.
- **Rare Word and OOV Handling:** Due to the absence of subword tokenization, rare and compound words were replaced with unknown tokens, leading to partial or missing translations.

To mitigate these issues, several potential improvements are suggested:

- Employing a sub word tokenisation scheme for richer morphological coverage and better rare-word handling.
- Using pretrained or adaptive embeddings aligned across the encoder and decoder to maintain semantic consistency and reduce incoherent generations.

4. Interesting Insights

A notable observation was that **pretrained embeddings** (FastText and GloVe) led to faster convergence and slightly higher BLEU scores compared to randomly initialized embeddings. This highlights the benefit of incorporating external semantic knowledge, which allows the model to learn meaningful representations even with limited bilingual data.

Another key insight was that **increasing model depth beyond three Transformer layers** led to slight degradation. This is likely due to overfitting on the relatively small parallel corpus.

Furthermore, **Hindi translations were more stable and accurate than Bengali translations**. This can be attributed to multiple factors:

- Bengali is morphologically richer than English, making it inherently more complex to model.
- The Hindi training corpus was larger, consisting of approximately 80,000 sentence pairs, whereas the Bengali corpus contained around 68,000, contributing to better generalization in the Hindi subtask.

These insights collectively emphasize the importance of data scale, linguistic similarity, and embedding initialization in achieving robust translation performance across diverse Indian languages.

8 Conclusion

In this project, we developed and systematically evaluated multiple neural machine translation models for English–Hindi and English–Bengali translation tasks. Starting from simple GRU-based encoder-decoder architectures, we progressed toward Transformer-based models that achieved substantial gains in BLEU, ROUGE, and chrF++ scores.

The final Transformer model, incorporating three encoder-decoder layers, multi-head self-attention, and pretrained 300-dimensional embeddings, achieved the best performance with a BLEU score of 0.172 on the test set.

Key findings:

- Attention mechanisms significantly improve translation quality by enabling selective focus on source tokens.
- Transformer architectures outperform RNN-based models, especially in handling long-range dependencies and maintaining fluency.
- Pretrained embeddings accelerate convergence and improve translation accuracy.

Future Directions:

- Integrating subword-level tokenization to address rare-word challenges and improve handling of morphological variations.
- Using pretrained aligned embeddings to enhance semantic consistency between the encoder and decoder representations, leading to better translation quality and faster convergence.

In conclusion, this work demonstrates how progressively refining architecture, embeddings, and training strategies can lead to competitive translation systems even in restricted and low-resource scenarios.

References

[1] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014.

[2] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," in *International Conference on Learning Representations (ICLR)*, 2015.

[3] A. Vaswani, N. Shazeer, N. Parmar, *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.