CS779 Competition: Machine Translation System for India

Shivam Pal
231110608
{pshivam23}@iitk.ac.in
Indian Institute of Technology Kanpur (IIT Kanpur)

Abstract

I've developed a machine translation system capable of translating sentences from seven Indian languages, which include Bengali, Hindi, Gujarati, Kannada, Malayalam, Tamil, and Telugu, into English. The system employs a transformer-based encoder-decoder model in combination with a sentence-piece (byte pair encoding) tokenizer. In our testing, we achieved a charF++ score of 0.474, a Rogue score of 0.478, and a Bleu score of 0.198 on the evaluation dataset. Our system secured the top position in the competition, ranking 1st.

1 Competition Result

Codalab Username: $S\_231110608$
Final leaderboard rank on the test set: 1
charF++ Score wrt to the final rank: 0.474
ROGUE Score wrt to the final rank: 0.478
BLEU Score wrt to the final rank: 0.198

2 Problem Description

The objective is to perform machine translation from seven Indian languages specifically Bengali, Hindi, Gujarati, Kannada, Malayalam, Tamil and Telugu into English. We can formally define it as following.
Let $D = \{(x^i, y^i)\}_{i=1}^N$ be a training set where $x^i = x^i_{1:n}$ is the sequence of length n (source language sentence) and $y^i = y^i_{1:m}$ is the sequence of length m (target language sentence). Where $x^i_j \in \mathbb{V}_s$ (source vocab set) and $y^i_j \in \mathbb{V}_t$ (target vocab set) and $1 \leq n, m \leq L$ where L is largest sequence length. Let also assume that $x^i \sim p_X$ and $y^i \sim p_Y$. Then for a given source sentence x its translation can be define as following.

$$\hat{y} = \underset{t}{\operatorname{argmax}}\, p_{\hat{\theta}}(t \mid x)$$

where $t \in T$ set of all possible sequences in target language. $\hat{\theta}$ can be approximated by maximizing the posterior of training data.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{N} \log p_\theta(y^i \mid x^i)$$

Quality of translation can be calculated via some quality function $Q(\hat{y}, y)$ e.g. charF++, ROGUE, BLEU.

3 Data Analysis

Source-English	# sents	avg tgt-len	avg src-len	max tgt-len	max src-len
Hindi	80797	17	19	257	216
Bengali	68848	17	14	100	84
Gujarati	47482	18	16	96	93
Kannada	46794	16	12	100	78
Malayalam	54057	15	11	107	108
Tamil	58361	17	13	2080	74
Telugu	44904	16	12	100	81

Table 1: Training data Statistics

We excluded sentences with a length exceeding 32 words from the dataset, which accounted for approximately 97% of the original dataset. The remaining dataset was then divided into an 80% portion for training and a 20% portion for validation through a random split. Furthermore, we eliminated any Unicode characters outside their respective language's defined range, with spaces being the sole exception. The specific Unicode ranges for each language are provided below.

Language	Hindi	Bengali	Gujarati	Tamil	Telugu	Kannada	Malayalam
Unicode Range	2304-2432	2432-2560	2688-2816	2944-3072	3072-3200	3200-3328	3328-3456

Table 2: Unicode Range of Indian Language

4 Model Description

Transformer

We implemented a encoder-decoder architecture for machine translation based on the paper *Attention is All you Need*[1]. We follow the base implementation from Pytorch[1] tutorial. Model architecture is shown in the figure1. Figure source[2].

[1] `https://pytorch.org/`

[2] `https://en.wikipedia.org/wiki/Transformer_(machine_learning_model)`

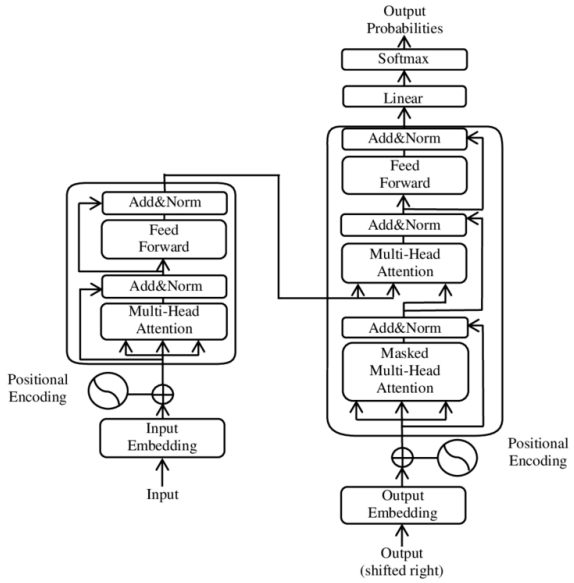

Figure 1: Transformer Architecture

Let K, Q, V be key, query and value matrices respectively the attention can be calculate as following.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V$$

5 Experiments

Tokenizer

We employed the sentence-piece[3] tokenizer, which is a language-agnostic tokenization tool. In our approach, we trained the source tokenizer using all source sentences, regardless of their language, and the target tokenizer using English sentences. Across various experiments, we explored different vocabulary sizes and model types, including byte-pair and unigram, to optimize our results.

Model	token	src vocab	tgt vocab	embd size	no. heads	Hidden dim	# decoder	# encoder
Model-1	Unigram	56K	8K	512	4	512	3	3
Model-2	Unigram	56K	8K	256	4	512	2	3
Model-3	Unigram	50K	10K	256	4	512	2	3
Model-4	Unigram	64K	16K	256	4	512	2	2
Model-5	BPE	64K	16K	256	4	512	2	2

Table 3: Models

Every model presented in the table was trained using Cross-Entropy loss, a learning rate of $1e-4$, and a batch size of 128. Model-4 and Model-5 underwent training with Cross-Entropy loss, and an additional label smoothing value of 0.3 was applied. To avoid overfitting, early stopping was implemented by monitoring both training and validation loss during training.

[3] `https://github.com/google/sentencepiece`

6 Results

Models	charF++ score	Rogue score	Bleu score	Decoding Strategy
Model-1	0.306	0.341	0.079	Greedy
Model-2	0.355	0.396	0.114	Greedy
Model-3	0.399	0.446	0.144	Greedy
Model-5	0.444	0.464	0.166	Beam-5
Model-5(LB)	**0.474**	**0.478**	**0.198**	Beam-5
Model-4(LB)	**0.476**	**0.481**	**0.196**	Beam-5

Table 4: Results

The results indicate that when model-5 and model-4 are trained with label smoothing, their performance is notably enhanced. Label smoothing serves as a method to discourage the model from being overly confident in its predictions. It mitigates the disparity between the highest probability label and other labels, which ultimately proves beneficial during the decoding process with beam search. It's evident that using label smoothing results in a significant improvement, as indicated by an increase in the Bleu score of 0.03.

7 Error Analysis

Our translation system has its flaws and often makes errors, yet it functions effectively when dealing with brief and straightforward sentences. However, its performance deteriorates when confronted with lengthy sentences. Among the various languages, it excels most in translating Malayalam.

8 Conclusion

We can deduce that the selection of vocabulary size plays a pivotal role in constructing an effective translation system. Opting for an excessively large vocabulary size might cause the model to overfit, while an overly small vocabulary size may result in numerous unknown (UNK) predictions. Furthermore, we have observed that implementing label smoothing is highly significant in mitigating overfitting, and it also offers advantages when dealing with beam search during the translation process.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems* (I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, eds.), vol. 30, Curran Associates, Inc., 2017.