Bangla Next-Word Prediction Using Bidirectional LSTM

Hafizur Rahman Sakib

September 22, 2025

1 Introduction

Next-word prediction is a key task in natural language processing (NLP), enabling applications such as text autocompletion, chatbots, and language modeling. This lab report details the development of a next-word prediction model for Bangla text, inspired by Ravjot Singh's blog, "Mastering Next Word Prediction with Recurrent Neural Networks (RNNs)" [**?**]. Unlike the blog's simple Recurrent Neural Network (RNN), this model employs a Bidirectional Long Short-Term Memory (LSTM) architecture to capture contextual dependencies in both directions, addressing limitations like vanishing gradients. The model was trained on a Bangla dataset describing a fictional city, Kankapur, and achieved a training accuracy of 99.98% with a loss of 0.0163 over 100 epochs. This report outlines the dataset, model architecture, training results, findings, and conclusions.

2 Dataset Details

The dataset, stored in an Excel file (`somoresh.xlsx`), contains 2,478 Bangla words across narrative and dialogue segments about Kankapur, a fictional city known for its peaceful environment and tourism appeal. The dataset has four columns: `segment_id`, `text`, `segment_type` (narration or dialogue), and `metadata` (contextual descriptions). The `text` column was used for training, preprocessed to retain only Bangla characters (`\u0980-\u09FF`), numbers, and punctuation (`.,?!`). The preprocessing yielded 2,473 sequences with a sequence length of 5 words and a vocabulary size of 1,168 unique words. Table **??** shows the first five dataset entries.

Table 1: First Five Entries of the Bangla Dataset

Segment ID	**Text**	**Segment Type**	**Metadata**
1	এরকম ঘটনা এই শহরে এর আগে ঘটেনি।	Narration	Introduction, setting the premise
2	তার আগে শহরটার পরিচয় দেওয়া দরকার। আমাদের চেনাশোনা আর পাঁচটা শহরের সঙ্গে এই শহরটির পার্থক্য হল এখানে আইন-শৃঙ্খলা সবাই মানে, বয়স্কদের শ্রদ্ধা করে কনিষ্ঠরা, কারণ এই শহরটিকে ওঁরা নিজেদের রক্ত দিয়েই তৈরি করেছেন বলা যায়।	Narration	Description of the city (Kankapur)
3	হিমালয়ের এই তল্লাটের আরও কিছু নামী-দামি শহর আছে যেখানে প্রতি বছর লক্ষ-লক্ষ মানুষ আসে □□□□□স্ট হয়ে।	Narration	Background: Tourism, reputation of water
4	সেই শহরে একদিন সকালে কাণ্ডটা ঘটে গেল।	Narration	Incident: A young man runs to the police station
5	যুবকুটি হাঁপাতে হাঁপাতে বলল, অফিসার, আমার স্ত্রীকে বাঁচান।	Dialogue	Young Man

3 Model Details

The model is a Bidirectional LSTM neural network designed to predict the next word in a Bangla text sequence. It improves upon the simple RNN in [**?**] by using LSTM layers to handle long-range dependencies and bidirectional processing to capture context from both directions. The architecture consists of:

- **Embedding Layer**: Maps 1,168 vocabulary words to 128-dimensional vectors, with an input length of 5 words.
- **Bidirectional LSTM**: 256 units, `return_sequences=True`, to process sequences bidirectionally.
- **Dropout**: 30% rate to prevent overfitting.
- **LSTM**: 256 units for further sequence processing.
- **Dropout**: 30% rate.
- **Dense (ReLU)**: 256 units for non-linear feature extraction.
- **Dense (Softmax)**: 1,168 units to output probabilities over the vocabulary.

The model was compiled with `categorical_crossentropy` loss, `adam` optimizer, and `accuracy` metric. It was trained on 2,473 sequences for 100 epochs with a batch size of 64, using `EarlyStopping` (patience=10, monitor `loss`) and `ReduceLROnPlateau` (patience=5, factor=0.5) callbacks. The pseudocode is shown below:

```
# Algorithm BanglaNextWordPrediction
# Input: Excel file with Bangla text, sequence length = 5
# Output: Trained model, tokenizer, predictions

# 1. Load Excel file using pandas.read_excel
# 2. Detect text column with Bangla characters (\u0980−\u09FF)
# 3. Clean text: keep Bangla characters, numbers, punctuation
# 4. Split text into words (2478 words)
# 5. Create sequences of length 5 (2473 sequences)
# 6. Tokenize sequences using Tokenizer (vocab_size = 1168)
# 7. Pad sequences (pre−padding) and one−hot encode outputs
# 8. Build model:
# − Embedding(vocab_size, 128, input_length=5)
# − Bidirectional(LSTM(256, return_sequences=True))
# − Dropout(0.3)
# − LSTM(256)
# − Dropout(0.3)
# − Dense(256, activation='relu')
# − Dense(vocab_size, activation='softmax')
# 9. Compile model: loss='categorical_crossentropy', optimizer='adam', metrics=['accuracy']
# 10. Train model: epochs=100, batch_size=64, callbacks=[EarlyStopping, ReduceLROnPlateau
    ]
# 11. Save model and tokenizer
# 12. For prediction:
# − Input text, clean and tokenize
# − Pad sequence to length 5
# − Predict top 3 words using model.predict and np.argsort
# − Return predicted words
```

4 Findings

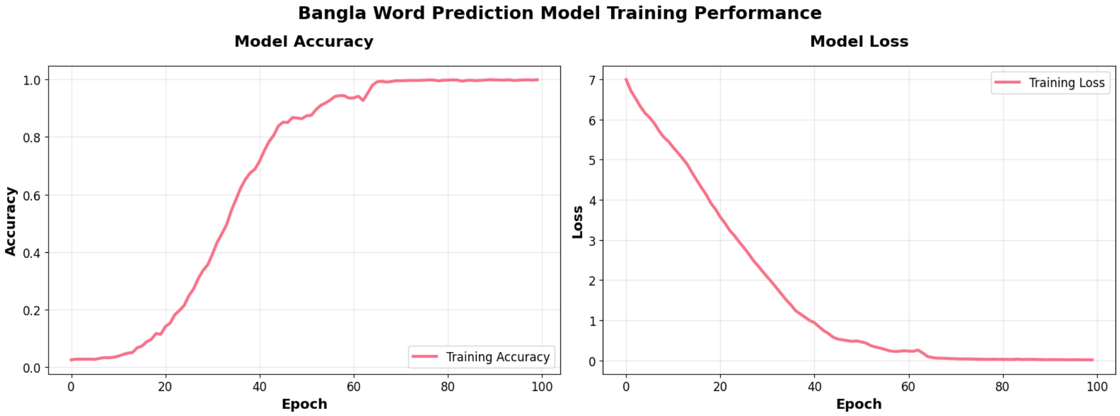
The model was trained on 2,473 sequences, achieving a final accuracy of 99.98% and a loss of 0.0163 after 100 epochs. The training took 57.56 seconds, with learning rate reductions at epochs 63 (to 0.0005) and 88 (to 0.00025) to optimize convergence. Key findings include:

- **Training Performance**: The loss decreased from 7.0359 (epoch 1) to 0.0163 (epoch 100), and accuracy increased from 1.67% to 99.98%, indicating effective learning of dataset patterns. Figure **??** shows the loss and accuracy curves.

- **Prediction Quality**: Interactive testing produced contextually relevant predictions in some cases (e.g., ``তারপর" for ``জমিটা পেয়ে গেছি। টাউন কমিটি স্যাংশন করেছে। শিগগির কাজ শুরু করব।"), but others were less accurate (e.g., ``কাছে" for ``হঠাৎ সুর্মা বলে উঠল, জানো বাবা,"), suggesting dataset bias or limited vocabulary diversity.

- **Limitations**: The high accuracy may indicate overfitting, as no validation split was used. The dataset's small size (2,478 words) limits generalization. Excluding the Bangla danda (``।") in preprocessing may have affected sentence boundary detection.

Figure 1: Training Loss and Accuracy Over Epochs



5 Conclusion

The Bangla next-word prediction model, built with a Bidirectional LSTM, successfully learned patterns from a 2,478-word dataset, achieving 99.98% accuracy and 0.0163 loss. The model's predictions were moderately accurate but showed limitations due to the small dataset and potential overfitting. The use of Bidirectional LSTM and dropout improved upon the simple RNN in [**?**], capturing contextual dependencies effectively. Future improvements should include a larger dataset, validation split, and pretrained Bangla embeddings to enhance prediction quality and generalization.

References

[1] Ravjot Singh, "Mastering Next Word Prediction with Recurrent Neural Networks (RNNs)," Medium, July 11, 2024. Available at: https://medium.com/@ravjot28/mastering-next-word-prediction-with-recurrent-neural-networks-rnns.