INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

RESEARCH ARTICLE

A Bi-LSTM and Attention Based Approach for Developing an Assamese AI Chatbot

Surajit Sarma[1]*, **Nabankur Pathak**[2]

1 Research Scholar, Department of Computer Science, Krishna Kanta Handiqui State Open University, Guwahati, Assam, India

2 Assitant Professor, Department of Computer Science, Krishna Kanta Handiqui State Open University, Guwahati, Assam, India

OPEN ACCESS

Received: 26-11-2023

Accepted: 09-09-2024

Published: 25-09-2024

Citation: Sarma S, Pathak N (2024) A Bi-LSTM and Attention Based Approach for Developing an Assamese AI Chatbot. Indian Journal of Science and Technology 17(36): 3811-3823. https://doi.org/10.17485/IJST/v17i36.3014

* **Corresponding author**.

suravsarma2007@gmail.com

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

Abstract

Objectives: The main objective of this study is to develop, evaluate, and analyses a Bi-LSTM and attention-based model and decide whether it is the best deep learning hybrid model for developing the Assamese AI Conversational Agent. **Methods:** A dataset for the Assamese language was created for this research. A bi-LSTM and attention-based model was developed and trained with the dataset. We have compared the accuracy and performance of the proposed model with those of other deep learning models. The bag of words technique was used for feature extraction of text. For natural language processing, a custom stemmer, stop word functions and root word and stop word database were developed. The performance of the model is measured using metrics such as precision, recall, and F1-score. **Findings:** From the research study, we have found that the bi-LSTM and attention-based model performed better in terms of all the metrics compared to other models. This model was able to attain an accuracy of 89.99%. **Novelty:** This research is novel as it is the first attempt to develop an AI chatbot in Assamese based on a deep learning model. Assamese is a language that has not received much attention in the field of AI chatbots. This research has great significance for the society where Assamese is spoken, as it can provide a platform for communication, information, and education in their native language. The Assamese language is in its beginning stages on the digital platform and has limited research done in the field of AI chatbots. This research will contribute a lot to this research area and open a new door for researchers who are interested in this topic. It will also allow them to further explore, improve, and develop new ideas.

Keywords: Deep Learning; Natural language processing; Chatbot; Assamese

1 Introduction

The development of new technology offers simpler and more effective interaction between humans and machines. Many digital assistant applications enable a user to give instructions to a computer to perform a particular function. Although there are several tools of this kind, the chatbot application is a particular one that is widely used by users.

The use of AI technology in chatbots has made them intelligent agents that are capable of performing tasks as per user needs. An AI chatbot is a software program that works like a virtual assistant in responding to the user's queries. AI offers many benefits, such as 24/7 information, multitasking, simultaneous service to many people, and reduced costs of infrastructure and manpower. These advantages have led many industries to adopt this technology for automated services to their clients. A research study was conducted by Mindbrowser in collaboration with Chatbot Journal, where they surveyed about 300 individuals from different sectors such as aviation, e-commerce, education, retail, etc. They have presented a graph where they show how different industries are using AI chatbots and benefiting from them.[1] The graph is presented in Figure 1.

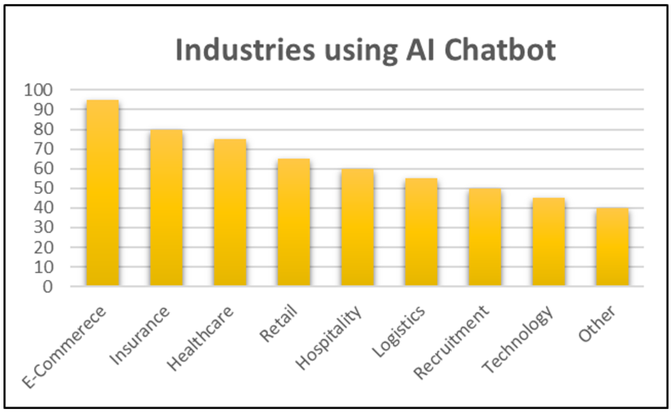

Fig 1. Industries benefiting from AI chatbots

As depicted in the figure above, e-commerce is the industry that most extensively employs AI chatbots to render services to its customers. Other industries that avail of AI chatbots include insurance, healthcare, retail, hospitality, logistics, recruitment, technology, and others. Chatbots have arisen as a popular and widely used platform for engaging and interacting with users in recent years and have attained immense popularity. AI chatbots are in high demand on digital platforms, and this has led to an emerging research field that explores their design, development, and evaluation.

According to[2,3], artificial intelligence (AI) chatbots use machine learning (ML) and natural language processing (NLP) techniques for imitating conversations between humans and computer. It produces relevant responses by understanding user intents and preferences. Platforms like voice assistants, chat apps, websites, and mobile apps are all possible integration points. AI chatbots have several benefits, including better customer experiences, lower operating costs, increased productivity, and the delivery of insightful analytics. These intelligent conversational agents go by a number of titles, such as virtual assistants, artificial conversational agents, or just conversational agents.

Natural language processing (NLP) is a tool used by specialized computer programs known as chatbots for interacting with humans. They use well-defined algorithms to produce appropriate responses for specified queries. Chatbots function as virtual assistants, providing individual support for diverse objectives, which include education or entertainment. According to research conducted by[4], chatbots are conversational intelligent computer programs that simulate human conversation in order to provide automated online assistance and guidance. According to[5], a chatbot is a piece of software that can assist and direct multiple users simultaneously. According to[6], chatbots—also known as chatterbots—are Human-Computer Interaction (HCI) systems that simulate human-computer conversations and are concurrently becoming increasingly popular.

The rise of AI chatbots has become a focal point in research, driven by technological advancements and the growing demand for educational resources in regional languages. Assamese, spoken by the majority in the northeastern part of India, particularly in the state of Assam, stands out as a language of interest. Creating an AI chatbot in Assamese poses a unique challenge compared to more widely recognized languages like English, Hindi, and Marathi. This challenge stems from the scarcity of resources and the lack of recognition for Assamese content on digital platforms, hampering the language's growth and accessibility in the digital realm. Furthermore, the field of AI chatbots in the Assamese language using deep learning remains largely unexplored, presenting a substantial challenge with limited existing research efforts.

Assamese is one of the Indo-Aryan languages of India. About 15-20 million people speak it in Assam, a state in northeast India. The Indian Constitution has declared that Assamese is the 22nd official language of India. Most Assamese speakers live in Assam, but some live-in other states near Assam. These states are Arunachal Pradesh, West Bengal, Manipur, Mizoram, Meghalaya, and Nagaland. Assamese has a complicated grammar. The order of words in a sentence is subject-object-verb. The grammar has many parts, like verbs, nouns, pronouns, genders, tenses, and more. The grammar does not have the same forms for everyone who speaks Assamese.

1.1. Related Work

Deep learning finds extensive application in Indo-Aryan languages, particularly in the realm of chatbots. In a notable example,[7] pioneered the development of a Bengali chatbot utilizing a transformer model for answering general knowledge questions. To enhance the model's proficiency, Masum curated a Bengali general knowledge question-answer dataset for training purposes, achieving an impressive 85.0 BLEU on the QA data. This chatbot demonstrated exceptional prowess in tackling sequence-related problems. Additionally,[8] contributed to the educational domain by constructing a Bengali chatbot using machine learning and BNLP, with the project named BIIB.[9] proposed an innovative approach employing deep neural networks to build chatbots trainable on diverse data, featuring text-to-speech functionality. In the healthcare sector,[10] recommended a hybrid model incorporating a knowledge graph and a text similarity model for the creation of a sophisticated healthcare chatbot capable of addressing intricate medical queries.

The majority of AI chatbot development that uses deep learning techniques was developed for the English language, with only a handful extending to Indo-Aryan languages like Bengali, Hindi, Urdu, and others. The Assamese language, however, presents a distinct structure compared to both Indo-Aryan languages and English. Consequently, models effective in English and other Indo-Aryan languages proved unsuitable for Assamese. Creating an AI chatbot in Assamese was a challenging task because there weren't many resources available for the public and there was no standard tool and data to use.

Several research papers have been published on the development of Information Retrieval (IR) systems for the Assamese language, characterized by limited natural language processing (NLP) tools and low resources. One study proposed an IR system utilizing the Vector Space Model and Assamese WordNet to enhance search results through query expansion with synonyms[11]. In a different approach,[12] presented a question-answering system employing a structured and logical representation of Assamese text to address natural language questions, contributing to advancements in AI and NLP for the Assamese language. Another contribution, outlined by[13] introduced an IR system utilizing Assamese WordNet and Wikipedia to organize queries and retrieve information from an extensive corpus. While the system demonstrated 60.08% performance in document retrieval, challenges arose with specific queries not covered by the Wikipedia corpus. The paper recommended enhancements, including the creation of a larger corpus, exploring multiple web sources for extensive queries, and implementing a more effective ranking mechanism such as the Vector Space Model. These studies collectively illustrate the evolving landscape and potential of IR systems for the Assamese language, offering benefits to users and researchers of this linguistic community.

This research endeavors to create an AI chatbot for the Assamese language through the application of deep learning techniques. Due to the absence of a standardized public dataset, we took the initiative to construct our own dataset tailored for the Assamese language. Our investigation revealed the absence of existing AI chatbots for Assamese that leverage deep learning methodologies. Consequently, we proposed a hybrid deep learning model incorporating Bi-LSTM and attention mechanisms, training it using our generated dataset. To gauge the accuracy and performance of our model, we opted for a comparative analysis with various models, including FFDN, RNN, LSTM, GRU, Bi-GRU, and Bi-LSTM. Assessment metrics such as recall, precision, and F1 score were employed to evaluate the performance of each model. In conclusion, we scrutinized and compared the outcomes of these models to identify the most effective one for our specific context.

The rest of the paper is divided into four sections. Section 2 describes the methodology. Section 3 presents the results and discussion. Section 4 concludes the paper. Section 5 presents references.

2 Methodology

In this section we have discussed the design methodology of the proposed system. The design methodology is divided into four parts. Figure 2 present the design methodology of proposed system.

2.1. Dataset

Constructing our own dataset became imperative due to the absence of a standardized dataset publicly available for the Assamese language. The data for our dataset was gathered from diverse sources, with the student support system of institutions, FAQs, interviews, and questionnaires serving as the primary contributors. Complementary data was extracted from secondary sources, such as the internet, encompassing emails and social media platforms like Facebook. To ensure compatibility, non-Assamese and English data were translated into the Assamese language by a dedicated team of translators. Our dataset comprises 2261 questions, each accompanied by its corresponding responses, and spans across 20 categories. The dataset has been

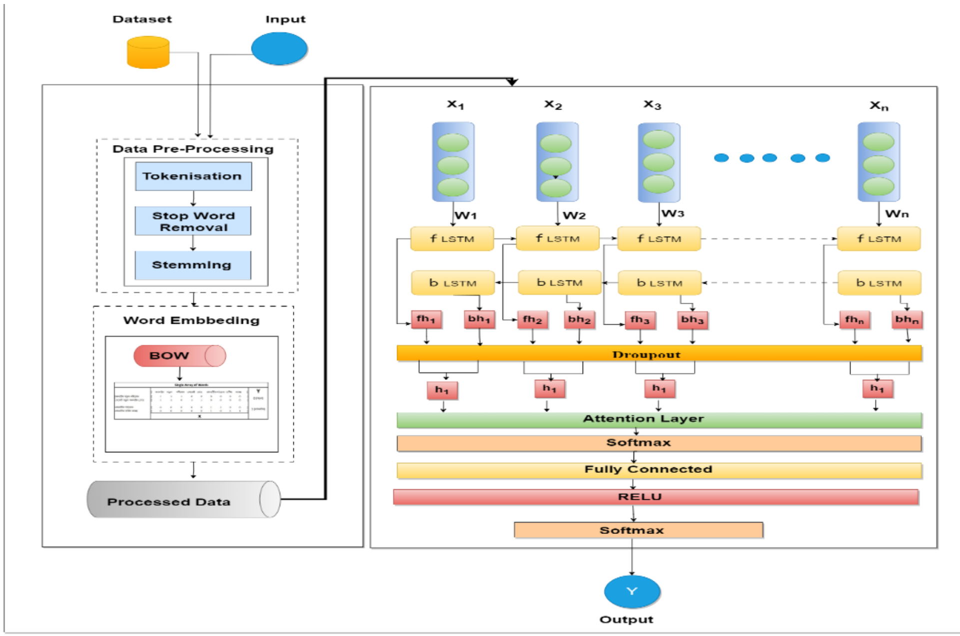

Fig 2. Proposed System Design Methodology

organized and stored in MySQL format. Table 1 illustrates the number of questions and answers per category.

Table 1. Number of questions and answers per category

Sl.no	Category- Meaning	No of Question Answer
1	পেমেন্ট - Payment	1372
2	নামভৰ্তি - Registration	283
3	বিষয়পৰিৱৰ্তন - Subject Change	37
4	শুধৰণি - Correction	50
5	সমাৱৰ্তন – Intermediate Semester	17
6	আপলোডকৰক - Upload	12
7	মাৰ্কশ্বীট - Marks Sheet	234
8	এডমিট - Admit Card	101
9	এচাইনমেণ্ট - Assignment	20
10	স্ব-শিক্ষণসামগ্ৰী - Self-learning material	35
11	কেন্দ্ৰপৰিৱৰ্তন – Study Centre Change	2
12	প্ৰৱেশদ্বাৰ - Entrance	13
13	প্ৰমাণপত্ৰ - Certificate	7
14	সন্মিলন – Counselling Class	1
15	পৰীক্ষা - Examination	45
16	পুনৰপৰীক্ষাকৰা - Re-examination	6
17	ফলাফল - Result	5
18	পৰামৰ্শ - Advice	3
19	যোগ্যতা - Eligibility	9
20	মাচুল – Fees Structure	9

2.2. Data Pre-Processing

In this segment, we provide a concise overview of the procedures employed to pre-process the dataset used for training the Assamese language AI chatbot. Utilizing the NLP Python package, we created an Assamese Natural Language Processing (ANLP) library, encompassing essential functions such as Assamese Word Tokenization, Assamese Stop Word Removal, and Assamese Word Stemming. These functions were systematically applied to both the dataset and user input queries. The subsequent section offers a brief delineation of the methodologies employed in implementing these pre-processing steps.

a) Assamese Word Tokenization: The concept of "Tokenization" involves breaking down sentences into meaningful, smaller units. In the development of an AI chatbot for the Assamese language, we engineered a tokenization function using the NLP Python library. This function effectively divides Assamese sentences into individual Assamese words.

b) Assamese Stop Word Removal: Assamese Stop Words, frequently used but less significant words, are systematically excluded from tokens in this process. Given the absence of a standardized stop word list for the public, we crafted our own, incorporating 116 stop words for the Assamese language. This list, derived from internet data, includes punctuations as stop words.

c) Assamese Word Stemming: The process of identifying the fundamental form of words, crucial for the chatbot to comprehend their meanings, involves the removal of prefixes and suffixes. To enhance the chatbot's understanding and responsiveness in Assamese, we created our own stemmer for the language. This involved compiling a dataset of 6,909 root words from various internet sources, as there is no publicly available standard stemmer for Assamese.

2.3. Word Embedding

To implement word embedding, we adopted the Bag of Words (BOW) technique. This method, rooted in machine learning, transforms textual content into numerical data. The approach entails tallying the frequency of each word within a sentence and constructing a table that catalogs these words alongside their respective counts. The initial step involves establishing a vocabulary encompassing all unique words present in the text. Subsequently, for every piece of text, a vector of the same length as the vocabulary is generated, initialized with zeros. As each word in the text is encountered, its index within the vocabulary is identified, and the corresponding element in the vector is incremented by one. This resultant vector serves as the feature representation of the text, facilitating the integration of textual data into numerical form.

2.4. Proposed model design

Bi-LSTM Layer: We employed the Bi-LSTM model to analyze sentences in the Assamese language. Bi-LSTM layer is a type of model that can handle sequential data, such as sentences. It has two components that process the data in opposite directions: forward and backward. This allows the Bi-LSTM layer to capture more information from the data and learn better. The Bi-LSTM model improves performance in these tasks by learning complex patterns and relationships between words in an Assamese sentence. This makes it a good choice for processing and understanding Assamese language data.

The forward LSTM (f LSTM) layer updates its hidden state h_t^f and its cell state c_t^f at each time step t by applying the following formulas:

$$i_t^f = \sigma(W_{ii}^f x_t + b_{ii}^f + W_{hi}^f h_{(t-1)}^f + b_{hi}^f) \tag{1}$$

$$f_t^f = \sigma(W_{if}^f x_t + b_{if}^f + W_{hf}^f h_{(t-1)}^f + b_{hf}^f) \tag{2}$$

$$g_t^f = tanh\left(W_{ig}^f x_t + b_{ig}^f + W_{hg}^f h_{(t-1)}^f + b_{hg}^f\right) \tag{3}$$

$$o_t^f = \sigma\left(W_{io}^f x_t + b_{io}^f + W_{ho}^f h_{(t-1)}^f + b_{ho}^f\right) \tag{4}$$

$$c_t^f = f_t^f \odot c_{(t-1)}^f + i_t^f \odot g_t^f \tag{5}$$

$$h_t^f = o_t^f \odot tanh\left(c_t^f\right) \tag{6}$$

where x_t is the input vector at time t , σ is the sigmoid function, $\odot$ is the element-wise product, and W and b are the weight matrices and bias vectors, respectively. The superscript f indicates the forward LSTM layer, and the subscripts i, f, g, and o indicate the input gate, forget gate, cell gate, and output gate, respectively.

The backward LSTM (b LSTM) layer takes the input sequence in reverse order and calculates the hidden state h_t^b and the cell state c_t^bat each time step t. It uses the same formulas as the forward LSTM layer, but with different weight matrices and bias vectors, which are called $W^b$and b^b. The letter b means the backward LSTM layer. The backward LSTM layer computes the hidden state h_t^b and the cell state c_t^bat time step t using the following equations:

$$i_t^b = \sigma\left(W_{ii}^b x_t + b_{ii}^b + W_{hi}^b h_{(t+1)}^b + b_{hi}^b\right) \tag{7}$$

$$f_t^b = \sigma\left(W_{if}^b x_t + b_{if}^b + W_{hf}^b h_{(t+1)}^b + b_{hf}^b\right) \tag{8}$$

$$g_t^b = tanh\left(W_{ig}^b x_t + b_{ig}^b + W_{hg}^b h_{(t+1)}^b + b_{hg}^b\right) \tag{9}$$

$$o_t^b = \sigma\left(W_{io}^b x_t + b_{io}^b + W_{ho}^b h_{(t+1)}^b + b_{ho}^b\right) \tag{10}$$

$$c_t^b = f_t^b \odot c_{(t+1)}^b + i_t^b \odot g_t^b \tag{11}$$

$$h_t^b = o_t^b \odot tanh(c_t^b) \tag{12}$$

The Bi-LSTM layer combines the hidden states of the forward and backward LSTM layers at each time step t. It does this by putting them together in one vector.

$$y_t = \left[h_t^f ; h_t^b\right] \tag{13}$$

where [;] denotes the concatenation operation.

Dropout Layer: In the proposed model, the dropout plays a crucial role by randomly excluding certain units in the network to counteract overfitting. Overfitting occurs when the model excessively adapts to the training data, hindering its ability to generalize effectively to new, unseen data. Dropout serves as a regularization technique, simplifying the model and enhancing its resilience to noise. The dropout layer is specifically employed on the output of the bidirectional LSTM layer before it undergoes the attention mechanism. Consequently, a subset of features in the LSTM layer's output is randomly set to zero, compelling the model to learn from a condensed representation of the input sequence. The mathematical formula for the dropout layer is:

$$o_i = \frac{b_i}{1 - rate} x_i \tag{14}$$

Where,

x_i is the input unit,

$b_i$is a binary mask that is 1 with probability $1 - rate$ and 0 with probability $rate$. It is sampled independently for each unit and each training step.

o_i is the output unit.

Attention Layer: The attention layer plays a pivotal role in learning to focus on the most pertinent segments of the bidirectional LSTM layer's output. This layer processes the concatenation of forward and backward hidden states by undergoing a linear transformation followed by a tanh activation function. Subsequently, another linear transformation is applied to generate a scalar value for each hidden state. These values are then subjected to normalization through a softmax function, yielding attention weights. These weights are instrumental in computing a weighted sum of the hidden states, constituting the ultimate output of the attention layer. This output is then directed to the fully connected layer for the purpose of classification.

The attention layer mentioned employs a query vector q, a key matrix L, and a value matrix as inputs M, generating an output vector P as the final result. The mathematical formulation for the attention layer in the aforementioned proposed model is outlined as follows:

$$Query\, Vector : q \tag{14}$$

$$Key\, matrix:\ L:\ \left[l_{1,}l_2, ...l_n\right] \tag{15}$$

$$Value\, matrix:\ M = [m_1, m_2, ...m_n] \tag{16}$$

$$Output\ vector:\ P \tag{17}$$

The attention layer initially computes the similarity or alignment score between the query vector and each key vector in the key matrix, utilizing a function aa such as dot product, cosine similarity, or a neural network. Subsequently, the alignment scores undergo normalization through a softmax function to acquire attention weights, indicating the contribution of each value vector to the output vector. The output vector is then computed via a weighted sum of the value vectors, proportionally scaled by the attention weights.

The initial step of the attention layer involves calculating the similarity or alignment score between the query vector and each key vector in the key matrix, employing a function a like dot product, cosine similarity, or a neural network. The alignment scores are then normalized by a softmax function to obtain the attention weights, which indicate how much each value vector contributes to the output vector. The output vector is then computed by a weighted sum of the value vectors, scaled by the attention weights.

The mathematical formula for the attention layer is:

Alignment score:

$$e_i = a\,(q, l_i) \tag{18}$$

Attention weight:

$$\alpha_i = \frac{exp(e_i)}{\sum_{j=1}^{n} exp(e_i\ )} \tag{19}$$

Output vector:

$$o = \sum_{i=1}^{n} a_i m_i \tag{20}$$

Fully Connected layer: The final component of the network is the fully connected layer, responsible for carrying out the classification task. It functions similarly to a linear layer, taking the context vector as input and producing an output vector. This output vector has a size referred to as ”num_classes,” indicating the number of distinct classes for the input sequence. The fully connected layer learns to map from the larger context vector to the smaller class vector. In simpler terms, it serves as the network segment that makes the ultimate decision regarding the classification task.

ReLU: After the fully connected layer, there’s a ReLU function. This makes sure that the result is always positive and adds a bit of complexity to the network. The formula for the ReLU function is:

$$f\,(x) = max(0, x) \tag{21}$$

Where,

x is the input

$f(x)$ is the output

Softmax: The softmax function is used as the neural network's final activation function in the proposed model. The goal is to confirm that the output result of the network is converted into a probability distribution over the expected result classes. This function makes sure that the total of the output values is 1 and that they fall within the range of 0 to 1. In this sense, the outcomes might be interpreted as the likelihoods of being in each class.

The mathematical formula for the softmax function is:

$$\sigma(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{k} e^{z_j}} \tag{22}$$

Where,

$\sigma(z)_i$ is the $ith\ component\ of\ the\ output\ vector,$

$$e\ is\ the\ mathematical\ constant,$$

$\sum_{j=1}^{K} e^{z_j}$ is the sum of the exponentials of all the input components.

2.5. Data Experiment and Training

- **Environment for experiment**

For this experiment, an Intel® Core^{TM} i5-10400 CPU @ 2.90GHz processor was employed, equipped with 6 cores, 12 logical processors, and 8.00 GB of RAM. The deep learning models were created using Python version 3.8.2 and PyTorch version 1.9.0.

- **Training Process**

In order to improve training of proposed model, we employ the Adam algorithm. It resembles a more intelligent variant of standard gradient descent. This approach accelerates the learning process with momentum by modifying the model's learning rate for each object it attempts to learn. In order to prevent overfitting, we feed the optimizer with information about the model, the learning rate and weight decay.

2.6. Testing

- **Split dataset**

After preparing the data, we have to split it into two sets: one set for training and the other for testing. Scikit-learn is a useful Python library that we use for this. An inbuilt tool in this library does this task by classifying 80% of the data for training and 20% for testing.

- **Compare with other models**

We did the testing in two ways: one to see how accurate the proposed model is (accuracy testing), and the other to measure its performance using things like precision, recall, and F1 score (performance testing). To see how well our model stacks up, we compared it to six other deep learning models: FFDN, RNN, LSTM, GRU, Bi-LSTM, and Bi-GRU. These models are briefly described in the following section:

I. FFDN: FFDN, or feed-forward neural network, is a type of artificial neural network. It doesn't use feedback loops and has three layers: input, hidden, and output. It learns from weights and biases, performs tasks like classification, regression, and pattern recognition. The input layer takes data, the hidden layer extracts features, and the output layer provides the final result based on the model design.

II. RNN: One kind of neural network that can learn from sequenced or pattern-containing data is the recurrent neural network (RNN). It works well for tasks like prediction, speech recognition, text analysis, and classification. RNNs can handle huge quantities of data in machine learning because they have internal memory that can store expanded sequences. LSTM and GRU are two variants of RNNs that we have briefly discussed in the following section.

III. LSTM: LSTM is a type of recurrent neural network (RNN) designed to learn from data with sequences or patterns. It can process data both forwards and backwards, improving its understanding of the content. The LSTM model consists of four parts: a forget gate, an input gate, an output gate, and a cell state. This work together to decide what information to remember or forget. LSTMs are great for tasks like classification, prediction, and speech recognition, especially when dealing with complex and lengthy data.

IV. GRU: GRU is a special kind of neural network that learns from data with patterns. It's good at remembering or forgetting information for a long time and can handle data moving both forward and backward. GRU has two parts, a reset gate and an update gate, working together to decide what to remember or forget. It's simpler and faster than LSTM, making it useful for tasks like classification, prediction, and speech recognition.

V. Bi-LSTM: Bi-LSTM is a model that learns from data with a sequence, like the order of words in a sentence. It has two parts that process the data from both the forward and backward direction. This helps Bi-LSTM gather more information and better understand the data. It's handy for language-related tasks, like understanding speech or text.

VI. Bi-GRU: Bi-GRU is a fast and simple bidirectional neural network. It only has input and forgets gates. It can quickly remember or forget things for a long time and process data in both directions. Bi-GRU is popular for tasks like natural language processing and speech recognition, and it's faster and simpler than Bi-LSTM.

3 Results and Discussion

3.1. Tools used for Testing

We applied the scikit-learn Python package to test the proposed model, as it provides a convenient way to assess the accuracy and performance of deep learning models. We used the classification_report function, which generates a summary of various metrics for each class label.

3.2. Metric and Formula for Testing

Accuracy: Accuracy is how well a model predicts the right answers. It is the number of correct predictions divided by the total number of predictions. We can measure accuracy to see how good a model is on a dataset. One way to measure accuracy is to use the accuracy_score function from the sklearn.metrics module. This function compares the real labels and the predicted labels of a dataset and gives the fraction or the number of correct predictions. The formula for accuracy is:

$$\text{Accuracy} = \frac{Number\ of\ Correct\ Predictions}{Total\ Number\ of\ Predictions} \tag{23}$$

To assess how well a classifier in scikit learn works, we can use three common metrics: precision, recall and F1 score.

Precision: Precision tells us the proportion of positive predictions that are correct. It measures how precise the classifier is when it assigns a positive label to a sample.

$$Precision = \frac{TP}{TP+FP} \tag{24}$$

Where,

TP=True Prediction

FP= False Prediction

Recall: Recall is a way of measuring how good a classifier is at finding all the positive cases. It is the fraction of positive cases that the classifier correctly identifies out of the total number of positive cases. The formula for recall is:

$$\text{recall} = \frac{\text{TP}}{\text{TP}+\text{FN}} \tag{25}$$

F1: F1 score is a measure of how well a model can identify relevant items. It combines precision and recall, which are two different aspects of accuracy. F1 score is calculated by taking the harmonic mean of precision and recall. The F1 score can range from 0 to 1, where 0 means the model is completely wrong and 1 means the model is perfect. The formula for F1 score is:

$$F1\ Score = \frac{2 \times precision \times recall}{precision + recall} \tag{26}$$

These metrics are calculated in two ways: macro and weighted. Macro means that the metric is computed by averaging the scores for each class, without considering the class imbalance. Weighted means that the metric is computed by averaging the scores for each class, but weighting them by the number of instances in each class.

Mean Squares error: The difference between the expected and predicted values is measured by the Mean Squares Error (MSE), a widely used statistic. It is a method to evaluate how accurately the model predict the outcome. A model which is less accurate can be identified by a higher MSE value, whereas a model that is more accurate has a lower MSE value. The mathematical formula for MSE is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{27}$$

where:

- n is the number of observations
- y_i is the actual value of the variable being predicted for the i-th observation
- $\hat{y}_i$ is the predicted value of the variable for the i-th observation

3.3. Accuracy Testing Results

Table 2. Accuracy of different AI chatbot models for Assamese language

Model	Accuracy
FFDN	89.21
RNN	89.71
LSTM	89.15
GRU	89.76
Bi-LSTM	89.65
Bi-GRU	89.82
Proposed Model (Bi-LSTM+Attention)	89.99

The Table 2 shows the performance of different neural network models. The models are evaluated using one metric: accuracy. The results show that the best model in terms of accuracy is proposed model, which achieved a score of 89.99. This means that proposed model correctly classified 89.99% of the instances.

In conclusion, the results suggest that proposed model is the most effective model for this classification task, as it performed the best on the accuracy metric. LSTM is the least effective model, as it performed the worst on the accuracy metric.

The model's accuracy on the training data, or the data used to modify the model's parameters during the learning process, shows up on the training accuracy graph. The model's accuracy on testing data which is used to assess the model's performance is displayed on the testing accuracy graph. The model is learning and generalizing well as the training and testing accuracy graphs are ideally high and near to one another. The training and testing accuracy graph is presented in Figure 2.

Table 3. Mean Square Error (MSE) of different AI chatbot modelsfor Assamese language

Models	Mean Square Error
FFDN	6.53
RNN	6.16
LSTM	6.91
GRU	6.24
Bi-LSTM	6.2
Bi-GRU	5.88
Proposed Model (Bi-LSTM+Attention)	5.67

3.4. Mean Square Error Testing Result

The Table 3 shows the performance of different neural network models. The models are evaluated using one metric: mean square error (MSE).

The results show that the best model in terms of MSE is proposed model, which achieved a score of 5.67. This means that the proposed model had the smallest error in predicting the values.

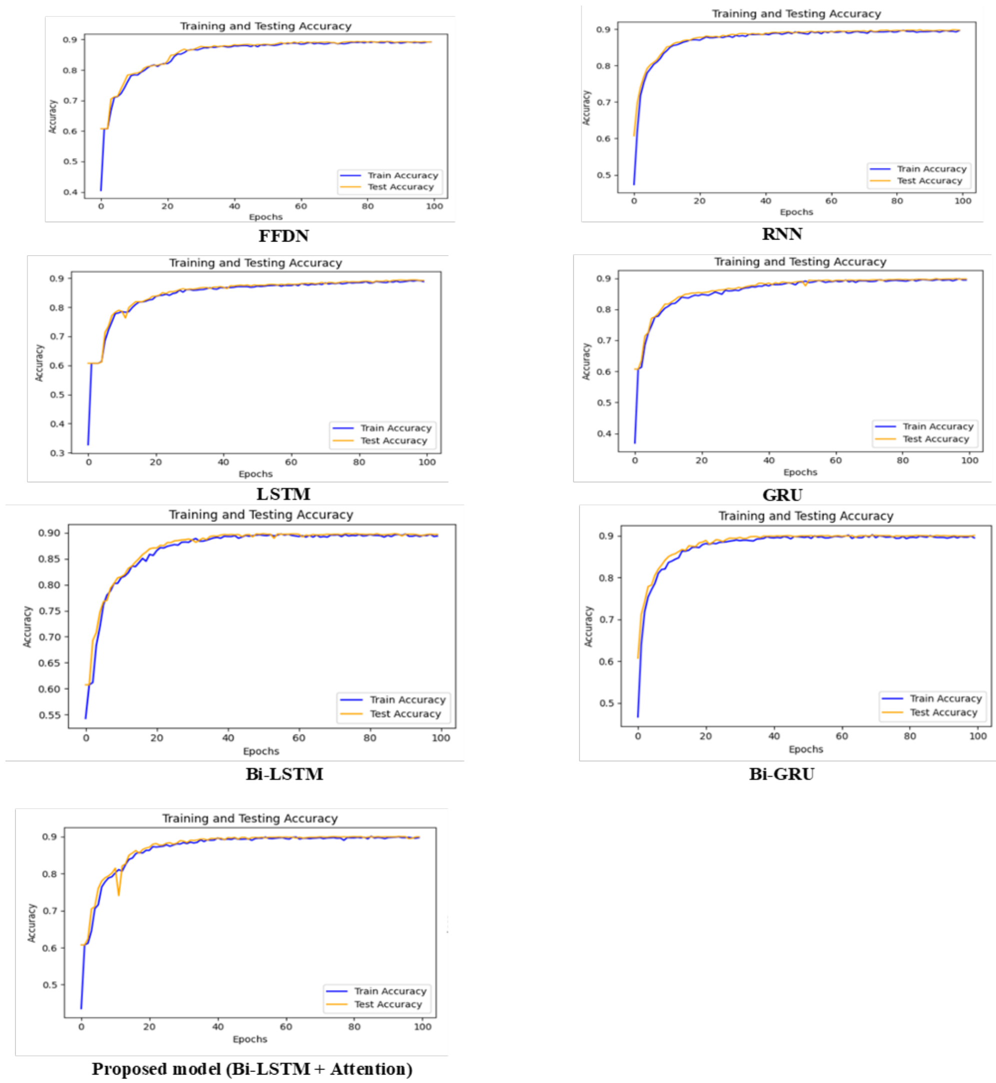

Fig 3. Training and Testing Graph of different AI chatbot models for Assamese language

In conclusion, the results suggest that proposed model is the most effective model for this regression task, as it performed the best on the MSE metric.

3.5. Performance Testing Results

Table 4. Precision, Recall, and F1 Score of different AI chatbot models for Assamese language

Model	Precision		Recall		F1 Score	
	Macro	Weighted	Macro	Weighted	Macro	Weighted
FFDN	0.77	0.89	0.7	0.89	0.72	0.88
RNN	0.94	0.9	0.78	0.9	0.83	0.89
LSTM	0.77	0.89	0.62	0.89	0.67	0.88
GRU	0.89	0.9	0.8	0.9	0.83	0.89
Bi-LSTM	0.94	0.91	0.78	0.9	0.82	0.89
Bi-GRU	0.94	0.91	0.8	0.9	0.84	0.89

Continued on next page

Table 4 continued

Proposed Model (Bi-LSTM+Attention)	0.94	0.91	0.8	0.9	0.84	0.89

The Table 4 shows the performance of different neural network models on some classification task. The models are evaluated using three metrics: precision, recall, and F1-score.

The results show that the best model in terms of macro precision are RNN, Bi-LSTM, Bi-GRU and the proposed model, which achieved a score of 0.94. This means that RNN, Bi-LSTM, Bi-GRU and the proposed model correctly classified 94% of the instances, regardless of their class. The best model in terms of weighted precision are Bi-LSTM, Bi-GRU and the proposed model which achieved a score of 0.91. This means that Bi-LSTM, Bi-GRU and the proposed model correctly classified 89% of the instances, but giving more importance to the classes with more instances.

The best model in terms of macro recall are GRU, Bi-GRU and proposed model, which achieved a score of 0.8. This means that GRU, Bi-GRU and proposed model retrieved 80% of the relevant instances for each class, regardless of their class. The best model in terms of weighted recall is GRU, Bi-LSTM, Bi-GRU, and the proposed model, which all achieved a score of 0.9. This means that these models retrieved 90% of the relevant instances for each class, but giving more importance to the classes with more instances.

The best model in terms of macro F1-score are Bi-GRU and the proposed model, which both achieved a score of 0.84. This means that these models had a good balance between precision and recall for each class, regardless of their class. The best model in terms of weighted F1-score is RNN, GRU, Bi-GRU, and the proposed model, which all achieved a score of 0.89. This means that these models had a good balance between precision and recall for each class, but giving more importance to the classes with more instances.

In conclusion, the results suggest the proposed model is the one that performed well on all the metrics.

3.6. Discussion

The best model depends on the evaluation metric that is used to measure the performance of the models. Different metrics may have different preferences and trade-offs. For example, accuracy is a simple metric that measures the overall correctness of the models, but it does not account for the class imbalance or the cost of misclassification. MSE is a metric that measures the error in predicting the values, but it does not account for the classification task or the distribution of the values. Precision, recall, and F1-score are metrics that measure the quality of the classification, but they can be calculated in different ways depending on the class weighting scheme (macro or weighted).

To summarize, the choice of the best model depends on the objective and the situation of the problem. We have shown that the proposed model (Bi-LSTM+Attention) is the best model, because it has the lowest MSE and the highest accuracy. Moreover, the proposed model has outperformed other models on metrics such as precision, recall and F1 score. Therefore, we can conclude that the proposed model (Bi-LSTM+Attention) is the best model for this task.

4 Conclusion

The purpose of this research was to create an AI chatbot for the Assamese language using a deep learning model that combines Bi-LSTM and attention. We made our own dataset and used it to train our model. We also created an Assamese NLP library that has our own stemmer and stopword remover, as well as other text processing functions. To see how our model compares with other models, we built six other models: FFDN, RNN, LSTM, GRU, Bi-LSTM and Bi-GRU and trained them on the same dataset. The test results show that our proposed model (Bi-LSTM + Attention) performed better than the other models on all metrics and achieved the highest accuracy of 89.99%. It was difficult to develop an AI chatbot for Assamese language because there are not many resources, standard datasets and tools such as stemmers or lemmatizers for this language. Our research is a new contribution in the field of Assamese AI chatbot based on deep learning, as no one has done it before. We hope this research will encourage more research in this field with new ideas.

Acknowledgement

The authors would like to express their sincere gratitude to Krishna Kanta Handiqui State Open University for providing me with the necessary support and help for carrying out this research and providing the data. Authors also appreciate the cooperation and assistance of the Assamese language speakers and the Assamese language translator team who participated in the data collection and evaluation. This research was not funded by any organization and was conducted as a part of academic interest. The authors hope that this research will contribute to the development and promotion of the Assamese language in the digital platform and the field of AI chatbots.

References

1) Global Chatbot Trends Report - 2020. 2020. Available from: https://www.mindbowser.com/industry-report/chatbot-survey/.
2) Chatbot Definition, Types, Pros & Cons, Examples. 2022. Available from: https://www.investopedia.com/terms/c/chatbot.asp.
3) Chatbots: What they are and why they're important. 2023. Available from: https://www.ibm.com/topics/chatbots.
4) Caldarini G, Jaf S, McGarry K. A Literature Survey of Recent Advances in Chatbots. *A Literature Survey of Recent Advances in Chatbots*. 2022;13(1):1–24. Available from: https://doi.org/10.3390/info13010041.
5) Costa P. Conversing with personal digital assistants: On gender and artificial intelligence. *Journal of Science and Technology of the Arts*. 2018;10(3):59–72. Available from: https://doi.org/10.7559/citarj.v10i3.563.
6) Madhumitha S, Keerthana B, Hemalatha B. Interactive Chatbot Using AIML. *International Journal of Advanced Networking and Applications*. 2019;(Special Issue):217–223. Available from: https://www.ijana.in/papers/62.pdf.
7) Masum AKM, Abujar S, Akter S, et al. Transformer Based Bengali Chatbot Using General Knowledge Dataset. In: 20th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE. 2022. Available from: https://doi.org/10.1109/ICMLA52953.2021.00200.
8) Kowsher M, Tithi FS, Alam MA. Doly: Bengali Chatbot for Bengali Education. In: 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT). IEEE. 2019;p. 1–6. Available from: https://doi.org/10.1109/ICASERT.2019.8934592.
9) Chopde A, Agrawal M. Chatbot Using Deep Learning. In: Proceedings of the International Conference on Innovative Computing & Communication (ICICC) 2022, Available at SSRN. 2022;p. 1–3. Available from: https://dx.doi.org/10.2139/ssrn.4140506.
10) Bao Q, Ni L, Liu J. HHH: An Online Medical Chatbot System based on Knowledge Graph and Hierarchical Bi-Directional Attention. In: ACSW '20: Proceedings of the Australasian Computer Science Week Multiconference. Melbourne. 2020;p. 1–10. Available from: https://doi.org/10.1145/3373017.3373049.
11) Barman AK, Sarmah J, Sarma SK. WordNet Based Information Retrieval System for Assamese. In: UKSim 15th International Conference on Computer Modelling and Simulation. IEEE. 2013;p. 10–12. Available from: https://doi.org/10.1109/UKSim.2013.90.
12) Chakraborty R, Sarma SK. Modelling a Question-Answering System using Structured Representation of Assamese Text. *International Journal of Artificial Intelligence & Applications*. 2015;6(6):1–10. Available from: http://dx.doi.org/10.5121/ijaia.2015.6601.
13) Bhuyan MP, Purkayastha R, Sarma SK, Sarmah S, Sarma P, Deka V. Information Retrieval In Assamese Using Wordnet & Assamese Wikipedia. *INTERNATIONAL JOURNAL OF SCIENTIFIC & TECHNOLOGY RESEARCH*. 2019;8(9):1887–1891. Available from: https://www.ijstr.org/final-print/sep2019/Information-Retrieval-In-Assamese-Using-Wordnet-Assamese-Wikipedia.pdf.