Elevating intelligent voice assistant chatbots with natural language processing, and OpenAI technologies

Nilesh B. Korade[1], Mahendra B. Salunke[2], Amol A. Bhosle[3], Gayatri G. Asalkar[4], Bechoo Lal[4], Prashant B. Kumbharkar[1]

[1]Department of Computer Engineering, JSPM's Rajarshi Shahu College of Engineering, Pune, India
[2]Department of Computer Engineering, PCET's, Pimpri Chinchwad College of Engineering and Research, Pune, India
[3]Department of Computer Science and Engineering, MIT Art, Design and Technology University, Pune, India
[4]Department of Computer Science and Engineering, Shri Jagdishprasad Jhabarmal Tibrewala University, Rajasthan, India

Article Info

Article history:

Received Apr 3, 2024
Revised Sep 10, 2024
Accepted Sep 29, 2024

Keywords:

Chatbots
Embedding
OpenAI
Particle swarm optimizer
Speech to text
Text to speech
XGBoost

ABSTRACT

Businesses can offer support to customers outside of usual business hours and across time zones by employing chatbots, which can provide round-the-clock support. Chatbots can react to user inquiries quickly, reducing wait times and improving customer satisfaction. It becomes challenging for the chatbot to differentiate between two queries that users pose that carry the same meaning, making it harder for it to understand and react appropriately. The aim of this research is to develop a chatbot capable of understanding the semantic meaning of questions as well as recognizing various speech patterns, accents, and dialects to provide accurate responses. In this research, we have implemented a voice-enabled chatbot system where users can verbally pose questions, and the chatbot provides responses through voice assistance. The architecture incorporates several key components: a question-answer database, OpenAI embedding for semantic representation, and OpenAI text-to-speech (TTS) and speech-to-text (STT) for audio-to-text and text-to-audio conversion, respectively. Specifically, OpenAI embedding is utilized to encode questions and responses into vector representations, enabling efficient similarity calculations. Additionally, extreme gradient boosting (XGBoost) is trained on OpenAI embeddings to identify similarities between user queries and questions within the dataset. This framework allows for seamless interaction between users and the chatbot, leveraging state-of-the-art technologies in natural language processing (NLP) and speech recognition. The outcome demonstrates that the XGBoost model delivers excellent outcomes when it is trained on OpenAI embedding and tuned with the particle swarm optimizer (PSO). The OpenAI-generated embedding has good potential for capturing sentence similarity and provides excellent information for models trained on it.

Corresponding Author:

Nilesh B. Korade
Department of Computer Engineering, JSPM's Rajarshi Shahu College of Engineering
Tathawade, Pune, Maharashtra, India
Email: nilesh.korade.ml@gmail.com

1. INTRODUCTION

Chatbots can save consumers time and effort by instantly responding to frequently asked questions on goods, services, regulations, and more. Chatbots have the capability to provide comprehensive details regarding products or services, encompassing features, specifications, pricing, and availability [1]. This aids customers in making well-informed decisions when considering a purchase. Chatbots are able to interact with consumers in the language of their choice, eliminating linguistic obstacles and ensuring accessibility for consumers with varying linguistic backgrounds. Chatbots have been crucial in transforming business procedures, interaction with clients, and communication [2]. Chatbots may save buyers time and effort by instantly responding to their most common questions on goods, services, norms, and more. Chatbots can provide quick advice on first-aid practices for common injuries and medical crises. They can offer precise guidance on how to handle situations like burns, cuts, fractures, and cardiac arrest. Compared to staff members, chatbots are extremely flexible since they can manage multiple interactions concurrently. This allows them to efficiently handle massive numbers of requests. Users can connect with chatbots in their preferred language because they can converse in various tongues. This reduces the need for professionals who are proficient in a particular language and enhances accessibility and user satisfaction, particularly for companies with international operations [3], [4].

Three significant obstacles lie ahead when developing a voice assistant chatbot. Accurate text transcription of spoken words is necessary for voice recognition technology. Voice-to-text conversion is affected by factors such as speech patterns, accents, and dialects, as well as low audio quality and background noise [5], [6]. Answering user inquiries that are semantically similar to those that are already in the dataset but asked in a different way can be one of the biggest hurdles in building a chatbot. If not handled properly, this problem may result in misinterpretations and inaccurate answers [7]. It can be challenging to build a chatbot that can answer clearly and naturally to a variety of audiences when it comes to text-to-speech (TTS) capabilities for accents and speech patterns, such as those seen in US and Indian English. There are variations in word pronunciation between Indian and US English. To assure correct pronunciation for every audience, the TTS system needs to take these distinctions into consideration. Finding two statements with the same meaning and response is the primary challenge in developing a voice assistant system. Queries like "Describe the process for registering an account with a bank" and "Explain the steps involved in opening a bank account" can be posed in a variety of ways by the user, and both have equivalent interpretations and replies. Such semantic relevance needs to be recognized by the assistance system [8].

The purpose of this research is to create a chatbot that can comprehend the semantic meaning of questions even when they use different words, as well as identify various speech patterns, accents, and dialects, to deliver accurate responses. Previous research has mostly focused on answering questions that could be successfully mapped. We have gathered common inquiries and their answers from a variety of sources, including first aid procedures, historical and geographical information, Ayurvedic home remedies, laws, mathematical concepts and formulas, recipes for Indian food. We preprocessed the data that had been collected and created a few new features based on textual properties. After examining several embeddings and training models to capture the semantic meaning and equivalency of both posed and dataset questions, we have decided to use extreme gradient boosting (Xgboost) and OpenAI embeddings. The user asks queries and receives a verbal response. OpenAI Whisper is used for transforming voice to text, and Tortoise TTS is used to translate text to speech. The finding demonstrates that the OpenAI-provided embedding places vector representations of questions with equivalent significance close to each other, making it easy for the model to recognize them. The experimental results demonstrate that XGBoost, trained using OpenAI embeddings and optimized with PSO, effectively captures the semantic meaning of questions even when different words are used as compared to state-of-the-art techniques. Additionally, OpenAI's TTS and STT systems can recognize various speech patterns, accents, and dialects to generate accurate responses for users.

2. LITERATURE SURVEY

People find it difficult to take seriously their health because of their hectic schedules, which results in fatal diseases which can harm people. An AI-powered healthcare chatbot system designed by Jegadeesan *et al.* [9] can recognize diseases and offer basic information about them. The chatbot provides you with Ayurvedic and homoeopathic remedies for associated health issues in addition to medical prescriptions and medications for those conditions. For every inquiry, the bot is trained to evaluate all possible answers, select the most relevant ones, and then deliver the results to the user. The study aims to make use of a database of well-organized responses and a retrieval-based approach that is based on directed flow theory or graph theory. Using natural language processing (NLP), the chatbot understands human input and uses the training data set to provide accurate responses. The suggested approach achieves an approximate 82% success rate.

Prayitno *et al.* [10] presented a chatbot that uses NLP to deliver health-related information. Preprocessing tasks, including lowercase, punctuation removal, and stopword removal, were handled by the NLP. When documents are embedded, a vector presentation of the same document with a closer vector is produced. The dot product of the vectors divided by the product of their lengths is cosine similarity, which is the cosine of the angle between the vectors. The study returns the response with the highest degree of similarity between the user's question and the answer found in the dataset using cosine similarity. The outcome demonstrates that the user's conditions were effectively evaluated by the medical chatbot, with an accuracy rate of about 87%.

In order to offer an alternative perspective to the conventional ways of conducting healthcare interviews, gathering symptoms, and making diagnoses, Vasileiou and Maglogiannis [11] constructed a health bot. The platform can analyze and categorize free text and voice input data into symptoms using NLP and speech recognition technologies, such as Google's DialogFlow. In order to estimate the possibility that a patient would have a certain disorder and to receive alerts in the event of a condition, the classified data has been used in the machine learning (ML) training process of the artificial intelligence (AI) models. One of the two trained logistic regression models is used to predict heart disorders, while the other is used to predict COVID-19 disease. The model has been trained and evaluated using the World Health Organization and Cleveland Heart datasets from UCI, yielding 98.3% and 82% accuracy, respectively.

The research objective of Zhou *et al.* [12] is to investigate whether it is feasible to use an artificially intelligent chatbot to promote COVID-19 vaccinations in a variety of healthcare settings. During the span of a year, from 2021 to 2022, the researcher implemented the system in the United States and recorded the number of users, the subjects that were discussed, and data regarding the correctness of the platform's replies to user desires. NLP contributes to analyzing and understanding user input messages, and the machine learning model is implemented to react to user requests. The result shows that when additional COVID-related information, like that regarding the Delta variant, appeared, accuracy dropped. The system's accuracy rate in matching responses to user queries ranged from 54% to 91.1%.

A rule-based chatbot was introduced by Rath *et al.* [13] to offer a brief overview of a person's mental state. Using Google Forms, this method gathered a dataset containing queries and responses from a wide range of participants. NLP is used for preliminary processing of the gathered dataset, including tokenization, stemming, lowercasing, punctuation removal, and missing value elimination. The text data is transformed into a vector with a vectorization technique like term frequency-inverse document frequency (TF-IDF). With the guidance of decision trees and established rules, these chatbots lead users towards specific behaviors.

El-Azhari *et al.* [14] constructed an inclusive chatbot that can deal with speech and text messages for learners with disabilities by using smart learning architecture. AI chatbots incorporate natural language understanding (NLU) to interpret user requests and natural language generation (NLG) to produce appropriate responses that offer accurate information. The suggested method automatically gathers pairs of questions and answers from a variety of discussion forums and learning resources, including Coursera, Guru99, Interview Bit, Simplilearn, Edureka, and Project Pro. The study tests four popular word embedding strategies for turning a sentence into a vector: Word2Vec, TF-IDF, GloVe, and FastText. Neural networks (NNs) are used in the proposed design for two primary reasons: first, they require less formal statistical training, and second, they have a high capacity to identify intricate nonlinear associations among dependent and independent variables. According to the results, Fasttext successfully classifies 70% of non-WH questions, while term frequency-inverse document frequency (TF-IDF) accurately classifies 90% of WH questions.

Using deep learning techniques, Anwar *et al.* [15] implemented an NLP-based chatbot for educational services that offers students and researchers comprehensive assistance. The proposed chatbot offers precise and informative university recommendations, question-answering (QA), and job search services by implementing sentiment analysis and sarcasm detection approaches. Using the Stanford Sentiment Treebank dataset, the pre-trained RoBERTa model has been enhanced for sentiment analysis. On the testing set, the RoBERTa model exceeded several state-of-the-art sentiment assessment models, such as bidirectional encoder representations from transformers (BERT) and XLNet, with an accuracy rate of 93.8%. The long short-term memory (LSTM) based bi-directional attention flow network (BiDAF) model outperformed earlier versions on comparable benchmarks, achieving an accuracy of 85% and an F1-score of 80% on a bespoke QA dataset.

Kothari *et al.* [16] implemented chatbots to carry out the electronic funds transfer (EFT) process for Turkish banking. The language descriptor layer determines whether the sentence entered by the user is in Turkish, and the intention recognition stage determines whether the sentence falls into one of three classes, such as EFT with account, IBAN, or credit card number. The pertinent sentence is sent to the named entity recognition stage, where information necessary for the EFT transaction, including the amount and account number, is obtained and the sentence's words are categorized based on their asset structures. The language detection process takes advantage of the Azure text language analysis rest API service. To extract the specified entities from the user-provided text, a multilingual BERT word embeddings library and a deep learning-based model have been used. The LSTM model has been used to provide responses based on the messages that users submitted to the chat. A 70% success rate was attained in EFT with account number, 90% in EFT with IBAN, and 90% in EFT with credit card number, according to the research communicated.

3. METHOD

The collected dataset containing questions and answers was first preprocessed, and various vectorization techniques were applied to transform it into numerical vectors. New features were created based on the dataset's properties, and dimensionality reduction was employed to decrease the vector size. Different models were analyzed using various vectorization techniques, and it was found that XGBoost, trained with OpenAI embeddings, performed the best. The optimal parameter values for training XGBoost were determined using the particle swarm optimizer (PSO) algorithm. Consequently, XGBoost was trained on the preprocessed dataset with the parameters identified by PSO, making the model ready to answer queries. OpenAI's speech-to-text (STT) system listens to the user's query, converts it to text, which is then preprocessed, vectorized, and fed to the model to find the relevant answer. The response is then converted back to speech using OpenAI's TTS system, providing a vocal response to the user's query. Figure 1 represents the strategies implemented in detail for building intelligent voice assistant chatbots.

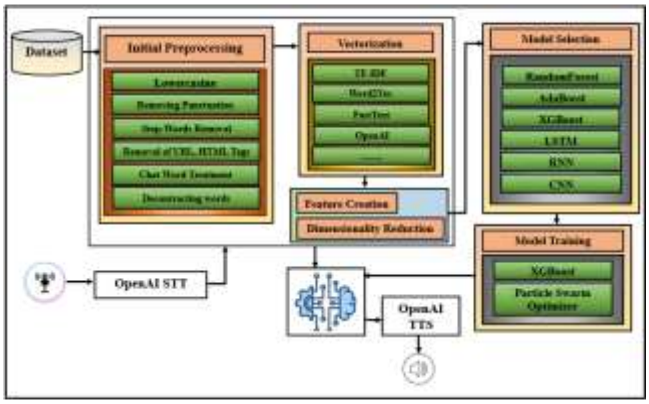

Figure 1. Implementation details for intelligent voice assistant chatbots

3.1. Dataset

The questions and answers in the dataset were gathered from a variety of sources, including guidelines on first aid processes, geographical and historical details, Ayurvedic home remedies, laws, mathematical formulas and concepts, Indian food recipes, and more. Table 1 displays the organization of the 10,000 samples that make up the dataset.

Table 1. Dataset

ID	Issue	Solution
01	I am having indigestion. What should I do?	Drinking warm water with lemon, carom seeds, and mindful eating are some of the Ayurvedic therapies for indigestion.
…	…	…

3.2. Preprocessing

Before the dataset can be used for vectorization and model training in the next phase, it needs to undergo basic preparation. The redundant or inaccurate records are checked out within the database, and those that are discovered are eliminated. Each phrase in the dataset is transformed to lower case; punctuation, URLs, HTML tags, and stop words have been eliminated; special symbols, such as the dollar symbol "$," have been replaced with their string equivalents; chart words, such as "ASAP," which stands for "as soon as possible," have been replaced; and lemmatization has been used.

3.3. Vectorization

Word embedding, also known as vectorization, is a technique that maps vocabulary words or phrases to an equivalent vector of real numbers. We have examined the performance of TF-IDF, Word2Vec, FastText, and OpenAI embedding for answering the corresponding questions.

3.3.1. TF-IDF

Term frequency (TF), which is the ratio of a word's occurrence in a document to the total number of terms in the document, is used to determine how frequently a word (w) occurs in a document (d). Within the entire corpus of documents D, the IDF shows the frequency or uncommonness of a word [17].

$$\mathrm{TFIDF(w,d,D) = TF(w,d)} \left[\frac{\text{frequency of w}'\text{s appearance in d.}}{\text{total number of words in document d.}}\right] * \\ \mathrm{IDF(w,D)} \left[\log \frac{\text{number of document in D.}}{\text{The number of documents that include w}}\right] \tag{1}$$

3.3.2. Word2Vec

The Word2Vec word embedding technique, which is based on the continuous bag-of-words (CBOW) and skip-gramme architectures, transforms words into vectors with closer vector representations for words with equivalent meanings. Contextual or surrounding words (X) are inputted into the CBOW in order to anticipate the target or middle word (Y), whereas the skip-gram architecture anticipates context or surrounding words (Y1, Y2, ...) for a given target word (X). The aim of skip-gram for a word sequence w_1, w_2... w_T can be represented as the average log probability describe in (2). Where v represents the target, v' represents the context vector representation of words, W represents the vocabulary size, and c represents the size of the training context. Applying the softmax function, the fundamental skip-gram formulation establishes this probability describe in (3).

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c\ j\neq 0} Log\ \ p(w_t + j \mid w_t) \tag{2}$$

$$P(W0|WI) = \frac{exp(v'_{wo}\ Tv_{wi})}{\sum_{w=1}^{W} exp(v'_{wo}\ Tv_{wi})} \tag{3}$$

In order to learn word embeddings and word classifications, Facebook AI research offers an open-source, free package called FastText. It generates vector representations so that a text's vector and the labels associated with it are equivalent. The softmax function is used by FastText to determine the probability of an appropriate label given its associated text, as described in (3), where S is the text that is associated with the label L [18].

$$F(W_L) = \frac{e^{w_S^T * w_L}}{\sum_{l\in L} e^{w_S^T * w_i}} \tag{4}$$

3.3.3. OpenAI

The approach of turning text into a vector of floating-point numbers is called an embedding. A vector's closeness is measured by its distance from another vector. Wide distances indicate low similarity, and minor variations indicate high relatedness. This serves as the foundation for OpenAI embeddings, which represent text using a type of NN known as a transformer. OpenAI presents three effective third-generation embedding models: "text-embedding-3-small", "text-embedding-3-large", and "text-embedding-ada-002". Common uses of OpenAI embeddings comprise searching, clustering, recommendations, identifying anomalies, diversity assessment and categorization. Text-embedding-ada-002 beats all preceding embedding models for text classification, and performs comparably for textual search, code search, and phrase similarity tests. Dealing with documents that are lengthy becomes easier by expanding the text-embedding-ada-002 context length by a factor of four, from 2048 to 8192 tokens. With only 1536 dimensions and a maximal input token of 8191, the new embeddings are only one-eighth the size of the davinci embeddings [7].

3.4. Feature creation

Robust features may boost machine learning models' forecasting capacity by providing relevant data. New features have been developed using the sentence characteristics that were listed in the dataset. Sentence length, number of words, ratio of common words to distinguished words, common word count, and distinct word count represent some of the fundamental features. The other features include average tokens, absolute length difference, first and last word equality, ratio of common stopwords to minimum and maximum stopword counts in both sentences, ratio of common word count to sentence length, and so on. Several strategies for string analysis and algorithm-based score of similarity computations, including fuzz ratio, fuzz partial ratio, token sort ratio, and token set ration were employed as feature functions in the fuzzywuzzy Python library [19].

3.5. Dimensionality reduction

The term "curse of dimensionality" demonstrates that the computational complexity of machine learning methods increases exponentially with the addition of dimensions, or characteristics, to the dataset. In order to eliminate the curse of dimensionality and transform data with high dimensions into lower-dimensional data, a feature extraction approach known as PCA can be used. Principal components, or the lines that contain the majority of the data's information, are the directions of the data that explain the greatest amount of variance. New variables built as linear combinations of the original variables are called principal components. Most of the information stored in the original variables is compressed into the first components as a result of these combinations, which result in uncorrelated new variables (i.e., principal components). As per PCA, the variance of a feature signifies its information content; the greater the variation in a feature, the more information it holds. The PCA explanation has been given here, step-by-step.

Standardize the continuous original variables' ranges to ensure that each one contributes equally to the analysis using (5). Standardization ensures that each variable has a standard deviation of one and a mean of zero.

$$Standardization\ (z) = \frac{Value(X) - Mean(\mu)}{Standard\ Deviation\ (\sigma)} \quad (5)$$

Covariance indicates the extent to which two or more variables fluctuate with respect to one another. Positive covariance indicates that the two variables are correlated, or increase or decrease together, while negative covariance indicates that the two variables are inversely correlated, or increase when the other declines. In (6) presents the expression for the covariance computation.

$$\text{Covariance}\ (x, y) = \frac{(x_i - \bar{x})/(y_i - \bar{y})}{N-1} \quad (6)$$

Where N is the total sample size, xi, yi are the data values, and x̄, ȳ are the means. Non-zero vectors that remain in the same direction after a linear transformation are known as eigenvectors. Assuming a "n × n" linear transformation matrix T and an eigenvalue λ, a non-zero vector X is an eigenvector if and only if it satisfies the following conditions conditions presented in (7).

$$\text{Tx} = \lambda\text{x} \quad (7)$$

Almost all vectors change direction after being multiplied by T. A few rare vectors known as eigenvectors indicate that x is pointing in the same direction as Tx. The eigenvectors of the covariance matrix serve as the primary axes of the data, and the projections of the data instances onto these principal axes are known as the principal components. The principal components can be obtained in order of significance by ordering your eigenvectors according to their eigenvalues, from largest to least [20].

3.6. Model selection

Several deep learning and machine learning models were trained with embeddings made using various techniques. By combining many decision tree forecasts instead of depending only on one tree's output, the random forest (RF) addresses the overfitting problem and high variance issues. To boost the efficiency of machine learning algorithms, AdaBoost, also known as "adaptive boosting," is an ensemble machine-learning technique that creates decision stumps using decision trees and combines weak learners into a single, powerful classifier. It builds the model by first giving each data point an identical weight. The data points whose classification by the previous model was wrong will be given more weight in the next iteration. In this way, it transfers model errors to subsequent models and attempts to enhance forecasting. The LSTM network is one variant of a recurrent neural network (RNN) that can recognize and detect order dependence in sequence prediction problems. The input, forget, and output gates are the three gates in the memory that regulate information flow and have the ability to add or remove data from the cell state [21]. 1D-CNN is a CNN version that specializes in the analysis of one-dimensional data sequences, such as words. The convolutional layers convolved the input, retrieved features from the input, and sent the output to the following layer by swiping the filter over the input matrix. The pooling layer attempts to progressively decrease the spatial dimension of the representation in order to minimize the number of parameters and computations in the network by multiplying the output matrix from the convolution layer and pooling matrix [22], [23].

3.6.1. XGBoost

Boosting is a ML ensemble approach that integrates weak classifiers to generate strong classifiers, which enhances their performance. Boosting works by training a series of weak models, typically decision trees, one after the other in an attempt to fix the mistakes of the earlier models depicted in Figure 2.

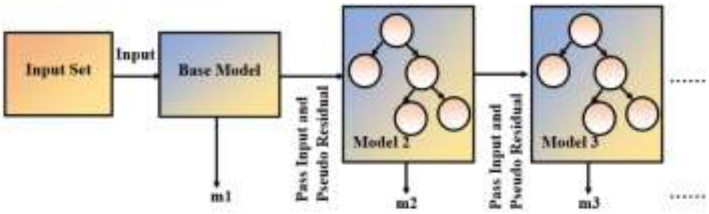
Input Set
Input
Base Model
m1
Pass Input and Pseudo Residual
Model 2
m2
Pass Input and Pseudo Residual
Model 3
m3
....
....

Figure 2. XGBoost architecture

A modified version of gradient boosting known as XGBoost involves parallel processing (at the node level) to boost speed and incorporates a number of regularization strategies to minimize overfitting and enhance overall performance [24]. Starting with the base model (m1), which calculates the log of odds given in (8), uses (9) to compute probability, and computes the pseudo residual using (10).

$$logit\ (P) = log\left(\frac{P}{1-P}\right) \tag{8}$$

$$Probability\ (P) = \frac{e^{logit\ (P)}}{1+e^{logit\ (P)}} \tag{9}$$

$$Pseudo\ R\text{esidual (R1)} = \text{Actual} - \text{Predicted} \tag{10}$$

Decision stumps are created by sending the input and computed residuals to a subsequent model. Once the data has been sorted, every residual is added to the leaf node, and (11) is used to determine the similarity score.

$$\text{Similarity Score (SS)} = \frac{(\sum Residual_i)^2}{(\sum(Previous\ Probability_i * (1 - Previous\ Probability_i) + \lambda))} \tag{11}$$

After splitting based on the average of two consecutive inputs, the left and right branches' similarity scores are determined, and finally the gain is estimated using (12). The splitting criterion that yields the highest gain is chosen.

$$Gain = \text{Similarity Score}_{Left} + \text{Similarity Score}_{Right} - \text{Similarity Score}_{Root} \tag{12}$$

Ultimately, the node output value is computed as expressed in (13), followed by the model prediction for the given input as expressed in (14).

$$\text{Node Output Value} = \frac{(\sum Residual_i)^2}{(\sum(Previous\ Probability_i * (1 - Previous\ Probability_i) + \lambda))} \tag{13}$$

$$Pediction = m1 + \ \lambda(m2) + \lambda(m3) \ldots .where\ \lambda\ is\ regularization\ term \tag{14}$$

After the computation of the residual, the residual and input are used to produce a decision stump for the next stage of model building. Until the residual reaches or equals zero, the process is repeated [25].

3.7. Optimization

PSO is a scientific computation approach designed to enhance a potential solution iteratively with respect to a given quality metric inspired by the social behavior of bird swarms. By feeding a potential solution into a model and accessing model performance against evaluation metrics, the optimization process attempts to raise or decrease the cost of an objective function [26]. Each particle in a PSO has three different variables: a velocity, which is a moving parameter; a fitness value determined by an objective function that takes the particle's position into consideration; and a position that correlates to a solution's attributes. The position of every particle in the swarm is modified in such a way that it gets closer to the particle with the optimal position. To update its position and velocity in each iteration, each particle keeps track of "gbest," the best solution found by all particles, and "pbest," the best solution it independently identified. Below is an overview of the stages of processing that are required to implement the PSO algorithm [27].

The following inputs are used: population size (n), number of dimensions (d), number of iterations (max_iterations), objective function f(X), variable boundary, inertia weight (W), correlation factors (C1, C2), and random number (r1, r2).
The outcome: the optimal parameters
Stage 1: To generate random position p and velocity v in all dimensions, use the mentioned calculation:

$$Particle\ Position\ (x_i) = lower\_bound + random\_no * (upper\_bound - lower\_bound) \tag{15}$$

$$\text{Velocity } (\text{v}_\text{i}) = lower\text{\_bound} + random\text{\_no} * (\text{upper\_bound} - \text{lower\_bound}) \tag{16}$$

Stage 2: Assign pbest=x_i and gbest=best position among all particles after calculating the objective function values f(xi), where i=1.n, for each particle.
Stage 3:
For iteration(itr)=1 to max_iterations
For i= 1 to n

After updating the particle's position and velocity using (17) and (18), determine the value of the objective function f(xi), and the pbest for the particle.

$$X_i^{itr+1} = X_i^{itr} + V_i^{itr+1} \tag{17}$$

$$V_i^{itr+1} = WV_i^{itr} + C_1 r_1^{itr}\left(P_{itr}^b - X_{itr}\right) + C_2 r_2^{itr}(P^q - X_{itr}) \tag{18}$$

Where, V_i^t is the velocity of the i^th^ particle at iteration t, X_i^t is the position of a particle, V_i^{t+1} and X_i^{t+1} are the newly calculated velocity and position. The P_t^b is the particle's best position for iteration t and P^q is the global best position.
End for i
Assign gbest = best position among all particles.
End for itr

3.8. Voice and text conversion

An automated speech recognition system called OpenAI Whisper has been trained with 0.7 million hours of multilingual, multitask-supervised web data. The implementation of an extensive and diverse dataset enhances robustness towards pronunciation, background noise, and technical terminologies. It also allows transcription in many languages and translation from those languages into English. An AI model called TTS transforms text into verbal speech that sounds natural. Two distinct model variates are: "tts-1," which is optimised for real-time TTS scenarios, and "tts-1-hd," which is optimised for quality [28].

4. RESULTS AND DISCUSSION

The experimental setup, performance evaluation of the proposed technique, and discussion of the findings are presented below.

4.1. Experimental setup

The implementation of this study was carried out using the Jupyter Notebook environment available on Google Colab. The Python library Pandas was utilized for efficient data manipulation, while NLTK and spacy were employed for initial pre-processing tasks. For vectorization, the study made use of Gensim, OpenAI tools, and scikit-learn library. Model building and visualization were conducted using scikit-learn, XGBoost, TensorFlow, and Seaborn library. A wide range of sources, such as first aid procedures, historical and geographical information, Ayurvedic home treatments, mathematical formulas and concepts, Indian food recipes, and more, were used to construct the 10,000 questions and answers dataset. Using the phrase's characteristics listed in the dataset, new features have been constructed. Several embedding techniques were used to train the various models on dataset. In order to evaluate the effectiveness of the proposed approach, 1000 questions with identical meanings but distinct text were generated.

4.2. Performance validation

Table 2 presents the various vectorization techniques and how effectively they work with different algorithms. The number of accurate and inaccurate responses generated by all models using the different embedding strategy is shown in Figures 3 and 4.

Table 2. Performance of different embeddings with various algorithms

Model	TF-IDF			Word2Vec			FastText			OpenAI		
	Answered correctly	Answered incorrectly	Accuracy	Answered correctly	Answered incorrectly	Accuracy	Answered correctly	Answered incorrectly	Accuracy	Answered correctly	Answered incorrectly	Accuracy
RF	787	213	78.7	806	194	80.6	804	196	80.4	812	188	81.20
Adaboost	792	208	79.2	834	166	83.4	838	162	83.8	844	156	84.40
XGBoost	861	139	86.1	908	92	90.8	912	88	91.2	923	77	92.30
LSTM	815	185	81.5	839	161	83.9	837	163	83.7	862	138	86.20
CNN	858	142	85.8	901	99	90.1	897	103	89.7	908	92	90.80

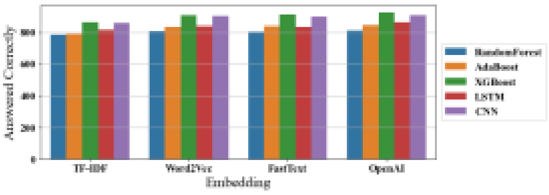

Figure 3. Number of accurate responses

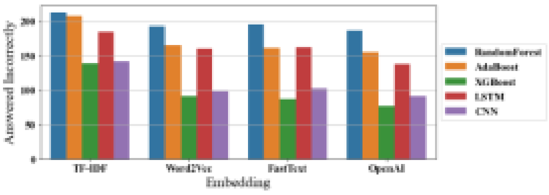

Figure 4. Number of inaccurate responses

4.3. Results discussion

The results show that OpenAI embedding has a great deal of potential for identifying the semantic meaning of questions asked, as it can improve chatbot performance by creating vectors of questions with similar meanings that are closer to one another. Comparing the XGBoost to CNN, LSTM, Adaboost, and RF, the XGBost demonstrates greater potential for correctly acknowledging the question. PSO optimisation is used for estimating the optimal hyperparameter value in order to further improve XGBoost performance. Results indicate that PSO finds the ideal XGBoost parameter only in the first iteration, and increases its accuracy by 2%. Table 3 presents the parameter's optimal value for XGBoost as well as their lower and upper bounds.

Table 3. XGBoost hyperparameters

Parameter	n_estimators	max_depth	learning_rate	gamma	eta
Lower bound	100	3	0.001	0	0
Upper bound	1000	10	0.3	5	1
Optimal value	150	5	0.004	1	0.01

5. CONCLUSION

Compared to human agents, chatbots are much more scalable since they can handle several requests at once, provide customer service outside of regular business hours, and function across time zones. The user can converse with the chatbot and receive vocal responses. The outcome reveals that OpenAI embedding builds vector representation in a way that makes questions with similar meanings closer to one another, improving the performance of machine learning models. XGBoost trained on OpenAI embedding accurately identifies the question and its corresponding response, and PSO determines the ideal parameter that improves XGBoost's forecasting performance. Even though a user may pose the same question in a different way, the chatbot should be able to discover its semantic meaning. The proposed approach shows promise in responding to queries with semantic meanings similar to those in the database. Furthermore, we can try different embeddings, algorithms, and optimisations in subsequent research to achieve better results. By remembering and referring to previous interactions over extended periods of time, voice assistants can be explored to facilitate meaningful and continuous conversations with users.

REFERENCES

[1] N. B. Korade *et al.*, "Exploring NLP techniques for duplicate question detection to maximizing responses on Q&A websites," *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 3, pp. 11–20, 2024.

[2] V. Oguntosin and A. Olomo, "Development of an e-commerce chatbot for a university shopping mall," *Applied Computational Intelligence and Soft Computing*, vol. 2021, pp. 1–14, Mar. 2021, doi: 10.1155/2021/6630326.

[3] S. H. Anwar, K. M. Abouaish, E. M. Matta, A. K. Farouq, A. A. Ahmed, and N. K. Negied, "Academic assistance chatbot-a comprehensive NLP and deep learning-based approaches," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 33, no. 2, pp. 1042–1056, Feb. 2024, doi: 10.11591/ijeecs.v33.i2.pp1042-1056.

[4] E. Adamopoulou and L. Moussiades, "Chatbots: history, technology, and applications," *Machine Learning with Applications*, vol. 2, p. 100006, Dec. 2020, doi: 10.1016/j.mlwa.2020.100006.

[5] C. Rzepka, B. Berger, and T. Hess, “Voice Assistant vs. chatbot – examining the fit between conversational agents’ interaction modalities and information search tasks,” *Information Systems Frontiers*, vol. 24, no. 3, pp. 839–856, Jun. 2022, doi: 10.1007/s10796-021-10226-5.
[6] A. Pande and D. Mishra, “The synergy between a humanoid robot and whisper: bridging a gap in education,” *Electronics (Switzerland)*, vol. 12, no. 19, p. 3995, Sep. 2023, doi: 10.3390/electronics12193995.
[7] K. I. Roumeliotis and N. D. Tselikas, “ChatGPT and Open-AI models: a preliminary review,” *Future Internet*, vol. 15, no. 6, p. 192, May 2023, doi: 10.3390/fi15060192.
[8] Z. Dong, Q. Ding, W. Zhai, and M. Zhou, “A speech recognition method based on domain-specific datasets and confidence decision networks,” *Sensors*, vol. 23, no. 13, p. 6036, Jun. 2023, doi: 10.3390/s23136036.
[9] R. Jegadeesan, D. Srinivas, N. Umapathi, G. Karthick, and N. Venkateswaran, “Personal healthcare chatbot for medical suggestions using artificial intelligence and machine learning,” *European Chemical Bulletin*, vol. 12, no. S3, pp. 6004 – 6012, 2023, doi: 10.31838/ecb/2023.12.s3.670.
[10] P. I. Prayitno, R. P. Pujo Leksono, F. Chai, R. Aldy, and W. Budiharto, “Health chatbot using natural language processing for disease prediction and treatment,” in *Proceedings of 2021 1st International Conference on Computer Science and Artificial Intelligence, ICCSAI 2021*, Oct. 2021, pp. 62–67, doi: 10.1109/ICCSAI53272.2021.9609784.
[11] M. V. Vasileiou and I. G. Maglogiannis, “The health chatbots in telemedicine: intelligent dialog system for remote support,” *Journal of Healthcare Engineering*, vol. 2022, pp. 1–12, Oct. 2022, doi: 10.1155/2022/4876512.
[12] S. Zhou, J. Silvasstar, C. Clark, A. J. Salyers, C. Chavez, and S. S. Bull, “An artificially intelligent, natural language processing chatbot designed to promote COVID-19 vaccination: a proof-of-concept pilot study,” *Digital Health*, vol. 9, Jan. 2023.
[13] S. Rath, A. Pattanayak, S. Tripathy, S. B. B. Priyadarshini, A. Tripathy, and S. Tanvi, “Prediction of a novel rule-based chatbot approach (RCA) using natural language processing techniques,” *International Journal of Intelligent Systems and Applications in Engineering*, vol. 11, no. 3, pp. 318–325, 2023.
[14] K. El-azhari, I. Hilal, N. Daoudi, R. Ajhoun, and I. Belgas, “An evolutive knowledge base for ‘AskBot’ toward inclusive and smart learning-based NLP techniques,” *International Journal of Advanced Computer Science and Applications*, vol. 14, no. 5, pp. 413–422, 2023, doi: 10.14569/IJACSA.2023.0140544.
[15] S. H. Anwar, K. M. Abouaish, E. M. Matta, A. K. Farouq, A. A. Ahmed, and N. K. Negied, “Academic assistance chatbot-a comprehensive NLP and deep learning-based approaches,” *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 33, no. 2, pp. 1042–1056, Feb. 2024, doi: 10.11591/ijeecs.v33.i2.pp1042-1056.
[16] S. Kothari, P. Bagane, M. Mishra, S. Kulshrestha, Y. Asrani, and V. Maheswari, “CropGuard: empowering agriculture with AI driven plant disease detection Chatbot,” *International Journal of Intelligent Systems and Applications in Engineering*, vol. 12, no. 12s, pp. 530–537, 2024.
[17] H. Vranken and H. Alizadeh, “Detection of DGA-generated domain names with TF-IDF,” *Electronics (Switzerland)*, vol. 11, no. 3, p. 414, Jan. 2022, doi: 10.3390/electronics11030414.
[18] Q. Du, N. Li, W. Liu, D. Sun, S. Yang, and F. Yue, “A topic recognition method of news text based on word embedding enhancement,” *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–15, Feb. 2022, doi: 10.1155/2022/4582480.
[19] N. B. Korade, M. B. Salunke, A. A. Bhosle, P. B. Kumbharkar, G. G. Asalkar, and R. G. Khedkar, “Strengthening sentence similarity identification through OpenAI embeddings and deep learning,” *International Journal of Advanced Computer Science and Applications*, vol. 15, no. 4, pp. 821–829, 2024, doi: 10.14569/IJACSA.2024.0150485.
[20] R. Drikvandi and O. Lawal, “Sparse principal component analysis for natural language processing,” *Annals of Data Science*, vol. 10, no. 1, pp. 25–41, Feb. 2023, doi: 10.1007/s40745-020-00277-x.
[21] N. B. Korade and M. Zuber, “Boost stock forecasting accuracy using the modified firefly algorithm and multichannel convolutional neural network,” *Journal of Theoretical and Applied Information Technology*, vol. 101, no. 7, pp. 2668–2677, 2023.
[22] N. B. Korade and M. Zuber, “Stock forecasting using multichannel CNN and firefly algorithm,” in *Cognitive Science and Technology*, vol. Part F1466, 2023, pp. 447–458.
[23] N. B. Korade and M. Zuber, “Forecasting stock price using time-series analysis and deep learning techniques,” in *Lecture Notes in Electrical Engineering*, vol. 1146 LNEE, 2024, pp. 403–421.
[24] L. Gan, “XGBoost-based e-commerce customer loss prediction,” *Computational Intelligence and Neuroscience*, vol. 2022, pp. 1–10, Jul. 2022, doi: 10.1155/2022/1858300.
[25] Z. Shao, M. N. Ahmad, and A. Javed, “Comparison of random forest and XGBoost classifiers using integrated optical and SAR features for mapping urban impervious surface,” *Remote Sensing*, vol. 16, no. 4, p. 665, Feb. 2024, doi: 10.3390/rs16040665.
[26] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of ICNN’95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948, doi: 10.1109/ICNN.1995.488968.
[27] N. B. Korade and M. Zuber, “Stock price forecasting using convolutional neural networks and optimization techniques,” *International Journal of Advanced Computer Science and Applications*, vol. 13, no. 11, pp. 378–385, 2022, doi: 10.14569/IJACSA.2022.0131142.
[28] L. Orynbay, B. Razakhova, P. Peer, B. Meden, and Ž. Emeršič, “Recent advances in synthesis and interaction of speech, text, and vision,” *Electronics (Switzerland)*, vol. 13, no. 9, p. 1726, Apr. 2024, doi: 10.3390/electronics13091726.

BIOGRAPHIES OF AUTHORS

Nilesh B. Korade is an Assistant Professor at JSPM’s Rajarshi Shahu College of Engineering, Pune, India. He is a data science professional with a Ph.D. in Computer Science and Engineering. His research areas are data science, data analysis, time series, machine learning, NLP, and deep learning. He has worked on several consultancy projects, filed several Indian patents, and obtained copyrights for his creative ideas. He can be contacted at email: nilesh.korade.ml@gmail.com.

Mahendra B. Salunke is an Assistant Professor at PCET's, Pimpri Chinchwad College of Engineering and Research, Pune, India. He is a data science professional with a Ph.D. in Computer Engineering. His research areas are data science, IoT, and microprocessors. He worked as Pune University's former department head and chairman of the microprocessor subject. He has worked on several consultancy projects, filed several Indian patents, and obtained copyrights for his creative ideas. He can be contacted at email: mahendra.salunke@pccoer.in.

Amol A. Bhosle is an associate professor at the School of Computing, MIT Art, Design, and Technology University, Pune, India. He is a data science professional with a Ph.D. in Computer Science and Engineering. His research areas are data science, data analysis, time series, machine learning, NLP, and deep learning. He has worked on several consultancy projects, filed several Indian patents, and obtained copyrights for his creative ideas. He can be contacted at email: amolabhosle@gmail.com.

Gayatri G. Asalkar is pursuing a Ph.D. from Shri Jagdishprasad Jhabarmal Tibrewala University, Rajasthan, India, and is an Assistant Professor at JSPM's Rajarshi Shahu College of Engineering, Pune, India. Her research areas are data science, data analysis, time series, machine learning, NLP, and deep learning. She has worked on several consultancy projects, published several research papers, and obtained copyrights for her creative ideas. He can be contacted at email: gayatri.teke@gmail.com.

Bechoo Lal is an Associate Professor at KL University, Andhra Pradesh, India. He holds a Ph.D. in Computer Science, a Ph.D. in Information Systems, and a PGP in Data Science from Purdue University, USA. He is a supervising Ph.D. research scholar from SJJT University, Rajasthan, and a member of the International Association of Engineers, USA. His research areas are data science, big data analytics, and machine learning. He can be contacted at email: blal2k7@gmail.com.

Prashant B. Kumbharkar is a Professor, department head, and dean of planning and design at JSPM's Rajarshi Shahu College of Engineering, Pune, India. He is a data science professional with a Ph.D. in Computer Science and Engineering. His research areas are data science, data analysis, time series, machine learning, NLP, and deep learning. He has worked on several consultancy projects, filed several Indian patents, obtained copyrights for his creative ideas, and guided several research scholars. He can be contacted at email: pbk.rscoe@gmail.com.