scientific reports

Check for updates

OPEN

Enhancing E-commerce recommendations with sentiment analysis using MLA-EDTCNet and collaborative filtering

E. S. Phalguna Krishna[1], T. Bhargava Ramu[2], R. Krishna Chaitanya[3], M. Sitha Ram[4], Narasimhula Balayesu[5], Hari Prasad Gandikota[6] & B. N. Jagadesh[7]✉

The rapid growth of e-commerce has made product recommendation systems essential for enhancing customer experience and driving business success. This research proposes an advanced recommendation framework that integrates sentiment analysis (SA) and collaborative filtering (CF) to improve recommendation accuracy and user satisfaction. The methodology involves feature-level sentiment analysis with a multi-step pipeline: data preprocessing, feature extraction using a log-term frequency-based modified inverse class frequency (LFMI) algorithm, and sentiment classification using a Multi-Layer Attention-based Encoder-Decoder Temporal Convolution Neural Network (MLA-EDTCNet). To address class imbalance issues, a Modified Conditional Generative Adversarial Network (MCGAN) generates balanced oversamples. Furthermore, the Ocotillo Optimization Algorithm (OcOA) fine-tunes the model parameters to ensure optimal performance by balancing exploration and exploitation during training. The integrated system predicts sentiment polarity—positive, negative, or neutral—and combines these insights with CF to provide personalized product recommendations. Extensive experiments conducted on an Amazon product dataset demonstrate that the proposed approach outperforms state-of-the-art models in accuracy, precision, recall, F1-score, and AUC. By leveraging SA and CF, the framework delivers recommendations tailored to user preferences while enhancing engagement and satisfaction. This research highlights the potential of hybrid deep learning techniques to address critical challenges in recommendation systems, including class imbalance and feature extraction, offering a robust solution for modern e-commerce platforms.

Keywords Sentiment analysis, Modified conditional generative adversarial network, Modified inverse class frequency algorithm, Ocotillo optimization algorithm, Attention-based encoder-decoder, Collaborative filtering

The last ten years have witnessed the meteoric rise of social media, which has permeated every aspect of society and revolutionized entire industries[1]. Among its numerous uses, social media facilitates two-way communication, the sharing of ideas and information, providing and receiving recommendations, and expressing personal opinions[2]. Today, before making a purchase, consumers not only consult those close to them but also scour the Internet for reviews, discussions, and other relevant information. Increasingly, businesses are embedding review and feedback sections directly into their websites and applications, enhancing transparency and convenience for both buyers and sellers[3]. Amazon, a well-known multinational corporation, exemplifies this trend by offering millions of products alongside customer ratings and reviews, aiding informed decision-making[4].

However, human intervention in summarizing and extracting insights from vast online reviews is challenging, making it difficult to identify relevant and useful information[5]. Sentiment analysis (SA), or opinion mining,

[1]Department of Computer Science and Engineering, GITAM School of Technology, GITAM University-Bengaluru Campus, Bengaluru, India. [2]Department of Electrical and Electronics Engineering, MLR Institute of Technology, Hyderabad 500043, Telangana, India. [3]Department of ECE, SRKR Engineering College, Bhimavaram, India. [4]Department of Computer Science and Engineering, School of Engineering and Sciences, SRM University, Amaravati, Andhra Pradesh, India. [5]Department of Computer Science and Engineering (AIML), Vasireddy Venkatadri Institute of Technology, Guntur, India. [6]Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Hyderabad 500075, Telangana, India. [7]School of Computer Science and Engineering, VIT-AP University, Vijayawada 522237, India. ✉email: nagajagadesh@gmail.com

addresses this challenge by processing unstructured text data to extract sentiments and predict polarity. Reviews regarding various goods and services, as well as massive amounts of unstructured data, can be transformed into structured data using this method[6]. Many varieties of SA exist; for example, some aim to determine whether reviews are positive or negative, while others seek to identify and categorize various emotions expressed in text.

With the growth of digital technology and ease of internet access, consumers are increasingly exposed to diverse information[7]. While this innovation broadens options, it also risks information overload due to the proliferation of sources[8]. Numerous methods have been developed to filter data before it is presented to users. One such method is recommendation systems, which aim to tailor information to each user's preferences, simplifying decision-making[9]. Recommender systems primarily utilize two types of information: explicit (e.g., ratings, comments, wish lists) and implicit (e.g., purchase history). Explicit data is directly provided by users, while implicit data is inferred from user actions[10].

In general, recommendation systems can be grouped into two primary types: collaborative filtering and content-based filtering[11]. Content-based approaches consider product attributes and group products based on content similarity[12]. However, this method often relies on external data, which may not always be available. Collaborative filtering, on the other hand, analyzes user behavior patterns within a neighborhood of similar users to suggest items[13]. Recommendations in collaborative filtering systems can be issued globally—providing identical suggestions to all users—or locally—offering personalized suggestions. Collaborative filtering systems use either memory-based or model-based methods. Memory-based approaches, which rely on user-item interaction data, provide higher accuracy for large datasets. Model-based methods are more efficient for smaller datasets[14,15].

It is essential to apply sentiment analysis (SA) techniques while developing a recommender system. SA, often called opinion mining, uses text mining and natural language processing (NLP) to extract sentiments or opinions from textual data[16]. These sentiments, which may be positive, negative, or neutral, are used to facilitate decision-making across industries[17]. SA-based recommendation systems incorporate sentiment polarity to enhance the accuracy of their suggestions. By integrating SA, recommendation systems can analyze user queries to predict positive, negative, or neutral sentiments and provide the most relevant results[18].

Understanding the role of social media and the concept of recommendation systems is critical in the present era[19]. By analyzing ratings, preferences, and user feedback, recommendation engines help consumers make informed choices, whether purchasing products, selecting careers, or choosing media content[20]. These systems play a crucial role in enhancing user interactions and driving business success.

Using deep learning (DL) practices, this research showcases a robust product recommendation system that combines SA with collaborative filtering (CF) to improve accuracy and performance. To address the class disparity problem, a Modified Conditional Generative Adversarial Network (MCGAN) is introduced, ensuring balanced minority class representation by generating synthetic data. Features are extracted using a Log-Term Frequency-based Modified Inverse Class Frequency (LFMI) algorithm, while a Multi-Layer Attention-based Encoder-Decoder Temporal Convolutional Network (MLA-EDTCNet) predicts sentiment. Finally, CF integrates sentiment insights to recommend over 25 items tailored to user preferences. Evaluation metrics such as accuracy, precision, recall, and F1-score validate the system's efficacy on an Amazon product dataset.

The key contributions of this manuscript are as follows:

- Novel Sentiment Analysis-Based Collaborative Filtering Framework: Unlike conventional models that treat sentiment analysis and recommendation separately, our framework integrates SA insights into CF for enhanced personalization.
- Addressing Class Imbalance Using MCGAN: Instead of standard oversampling or under-sampling approaches, we leverage a Modified Conditional Generative Adversarial Network (MCGAN) to generate synthetic samples, ensuring a balanced dataset.
- Development of MLA-EDTCNet: We introduce a novel Multi-Layer Attention-Based Encoder-Decoder Temporal Convolutional Network (MLA-EDTCNet) for sentiment classification, outperforming conventional deep learning models.
- Optimization with OcOA: Unlike standard tuning methods, we employ Ocotillo Optimization Algorithm (OcOA) to dynamically adjust hyperparameters and improve model efficiency.
- Improved Recommendation Accuracy: Our extensive experiments on an Amazon dataset show that our framework significantly outperforms state-of-the-art models in accuracy, precision, recall, and AUC.

The paper is structured as follows: section "Related work" reviews related work and research gaps, section "Proposed methodology" details the methodology, section "Results and discussion" discusses results, and section "Conclusion and future work" concludes the study.

Related work

Combining sentiment analysis (SA) with collaborative filtering (CF), Thomas and Jeba[21] present a new paradigm for product recommendation. They used an LSTM-based model for conducting SA and constructed two separate recommendation schemes using CF. The integration of the proposed SA model enhanced the best-performing recommendation system. The results demonstrated that the suggested methodology for product suggestions outperformed previous approaches. The combination of CF and SA has the potential to enhance product recommendation systems and improve customer satisfaction in e-commerce.

To perform SA on online product reviews, Rasappan et al.[22] proposed an improved machine learning (ML) technique called Enhanced Golden Jackal Optimizer-based Long Short-Term Memory (EGJO-LSTM). This approach involves four main steps: collecting reviews using a web scraping tool, preprocessing the data to refine it, applying term weighting and feature selection using IGWO and Log-term Frequency-based Modified

Inverse Class Frequency (LF-MICF) algorithms, and training the EGJO-LSTM model with the processed data. The model classified reviews as positive, neutral, or negative. Their experiments, conducted on an Amazon cloud dataset, showed that EGJO-LSTM outperformed conventional and hybrid approaches by 25% and 32% in terms of accuracy and precision, respectively. Additionally, using the LF-MICF technique, the EGJO-LSTM demonstrated superior performance compared to state-of-the-art methods.

Qin et al.[23] proposed a data-driven approach for product rating using SA and interval type-2 fuzzy sets (IT2FSs). Explicit attributes were extracted from reviews using a website, and implicit attributes were extracted using the Latent Dirichlet Allocation (LDA) model. The SA findings were represented as IT2FSs and classified using a deep learning model. Using the exponential TODIM (ExpTODIM) approach, they determined a ranking order after type-reduction for IT2FSs. A case study of Trip.com Group's travel product rankings based on user reviews demonstrated the effectiveness of the suggested strategy.

Wasi et al.[24] addressed challenges in Bangla sentiment analysis, including dataset limitations and linguistic subtleties, by proposing a model that integrates LSTM, Bi-LSTM, and CNN for enhanced text sequence classification. They evaluated six neural network models: ANN, CNN, LSTM, Bi-LSTM, BERT, and RCNN. The study highlighted that Bangla sentiment analysis is in its early stages and emphasized the need for improved algorithms and larger datasets to maximize its potential.

Dey et al.[25] proposed a Hybridized Deep Neural Network (HDNN)-based framework for sentiment analysis, optimizing feature space using a specially developed SentiWordNet. They adjusted Dispersive Flies Optimization to improve optimization and avoid local optima. Features were extracted using pre-trained embedding techniques and fed into hybridized convolutional representations and Long Short-Term Memory networks for opinion classification. The hybrid approach outperformed state-of-the-art methods on Amazon and Wine datasets, achieving accuracy rates of up to 91.3%.

Goularte et al.[26] introduced SentPT, a tailored polarity classifier for multi-genre sentiment analysis. Using a transfer learning approach to fine-tune a BERT model, they evaluated SentPT on diverse datasets, including product reviews, stories, and game comments. The classifier outperformed others in recall, precision, and F1-score, showing effectiveness across formal and informal text genres.

Danyal et al.[27] compared XLNet and BERT for natural language processing tasks. While BERT employed a masked language modeling objective, XLNet used a permutation language modeling objective. XLNet performed best on the IMDB dataset in terms of accuracy, demonstrating that sentiment analysis can predict movie review performance by identifying underlying attitudes and emotions.

Wu et al.[28] proposed an improved Markov model for recommendation systems. Enhanced state transition mechanisms and the Hidden Markov Model (HMM) boosted the model's predictive power for user behavior sequences. Experimental results using publicly available datasets showed that their model outperformed traditional approaches in user satisfaction and suggestion accuracy.

Iftikhar et al.[29] modeled recommendation as a sequential problem using Markov Decision Processes (MDPs). They developed a bi-cluster merging technique and applied reinforcement learning to find optimal recommendations. Their approach outperformed clustering methods, pure CF, and other reinforcement learning-based systems in policy learning, recall, and precision.

Sun et al.[30] proposed a CNN model with a gating mechanism for aspect-level SA. They combined fuzzy multi-attribute decision-making with predefined sentiment aspects, applying the model to a supplier selection dataset. Their solution outperformed state-of-the-art methods and provided more thorough supplier evaluations.

Nazari et al.[31] introduced a multi-agent recommendation system grounded in social network analysis. By generating user-item graphs and applying community detection and link prediction algorithms, the system clustered users and items effectively. Simulation results showed that their approach outperformed modern methods across multiple performance metrics.

Research gap

Despite the advancements in integrating sentiment analysis (SA) with recommendation systems, several challenges remain unaddressed. Many existing models, such as LSTM and CNN-based architectures, fail to effectively handle class imbalance issues, which can result in biased sentiment predictions. While techniques like Enhanced Golden Jackal Optimizer-based Long Short-Term Memory (EGJO-LSTM) and IT2FSs provide improvements in feature extraction and classification, they often lack scalability when applied to large and diverse datasets. Moreover, existing methods frequently overlook the dynamic interplay between explicit and implicit user preferences, limiting their ability to deliver highly personalized recommendations.

Another significant gap lies in the optimization of deep learning-based recommendation models. Current approaches inadequately balance exploration and exploitation, leading to premature convergence or inefficiency in high-dimensional data spaces. Additionally, while hybrid systems combining SA and collaborative filtering (CF) show promise, the integration often does not fully exploit the potential of advanced neural network architectures or metaheuristic optimization algorithms.

Lastly, the explainability and diversity of recommendations are underexplored. Future works need to focus on addressing these gaps by leveraging novel optimization techniques, robust neural architectures, and better handling of class imbalance to enhance the accuracy and reliability of recommendation systems.

Proposed methodology

SA is the core component of the proposed product suggestion engine, setting our approach apart from traditional methods by integrating a hybrid deep learning model (MLA-EDTCNet) with collaborative filtering (CF) to improve recommendation precision. Unlike previous works that separately handle sentiment analysis and product recommendation, our framework directly incorporates sentiment polarity into CF models, leading

to personalized product suggestions. Furthermore, our model addresses key challenges in sentiment analysis and recommendation systems, including:

- Handling class imbalance effectively through the Modified Conditional Generative Adversarial Network (MCGAN), which generates synthetic samples to balance sentiment classes.
- Leveraging a multi-layer attention mechanism in MLA-EDTCNet for better sentiment feature extraction, capturing contextual dependencies more effectively than conventional LSTM-based approaches.
- Employing the Ocotillo Optimization Algorithm (OcOA) for fine-tuning model hyperparameters dynamically, preventing premature convergence and improving recommendation accuracy.
- Demonstrating superior performance against state-of-the-art models in accuracy, precision, recall, and AUC, as validated on an Amazon product dataset.

A block diagram illustrating the proposed approach is shown in Fig. 1.

The proposed plan of action consists of two main components. The first is a GRU-based sentiment analysis (SA) model, and the second is the product recommendation system. Preprocessed data is used to train the SA model, which supports the development of collaborative filtering (CF)-based recommendation systems that are either user- or item-based. After selecting the best recommendation system, it utilizes user ratings to propose 20 products that they are most likely to purchase. Finally, to enhance the recommendations, the recommendation system is integrated with the sentiment analysis model. Based on an analysis of the review sentiments of the 20 suggested products, the model selects the top five.

Dataset

The dataset used in this study[23] was obtained from Kaggle. It includes more than 200 products and over 30,000 ratings from Amazon. The dataset encompasses a wide range of products, with more than 20,000 users providing ratings and reviews. Table 1 presents the attribute descriptions of the dataset in tabular form. Only the selected attributes are used for analysis by the SA model.

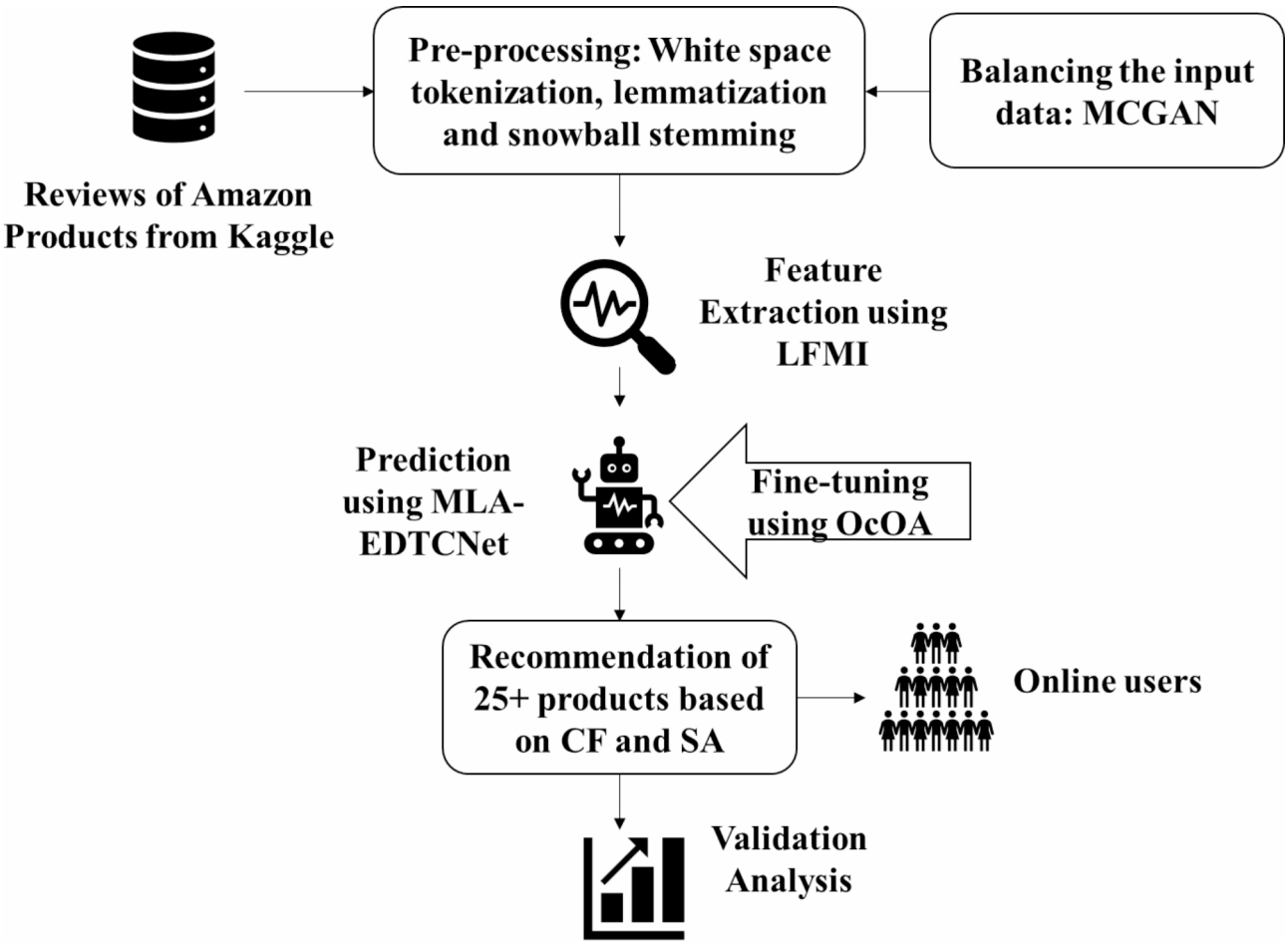

Fig. 1. Workflow of the proposed approach.

Attributes	Attribute description
Id	Each user evaluation of a product in the collection is associated with a unique identifier.
brand	The user's review besides rating for the product's brand
categories	Products in this category include things like books, personal care goods, medications, cosmetics, electrical appliances, health care, kitchen and dining, and a plethora of others.
manufacturer	Name of manufacturer of artefact
name	Name of the product to which user has added assessment or rating
reviews_date	When the user first other the review
reviews_didpurchase	The product's purchase status for a certain user
reviews_doRecommend	The sum of product recommendations made by a specific individual
reviews_rating	Rating given by user to an individual product
reviews_text	Customer feedback on a certain product
reviews_tittle	The user-generated review title for a certain product
reviews_userCity	The residing city of the user
reviews_userProvince	The exist in province of the user
reviews_username	A user's distinct identifier within the dataset
user_sentiment	The general opinion of customer regarding a specific product

Table 1. Dataset attributes.

Pre-processing

To filter out unwanted data, preprocessing techniques are applied. In this scenario, these steps are performed to prepare the data for sentiment analysis. The three preprocessing operations include "white space tokenization," "lemmatization," and "snowball stemming."

Tokenization

This method detects characters and breaks the text into words or tokens to understand the context and construct the natural language processing model. It further analyzes the word order to determine the meaning of the text.

Lemmatization

Lemmatization is a process that identifies the base form (lemma) of a word using vocabulary and morphological analysis. Additionally, it generates the structural dictionary of the word. Morphological analysis is employed to identify potential relationships in a multidimensional set and to solve problems involving multiple dimensions. Lemmatization converts the lemma into a standardized format, akin to a byte sequence. A lexeme represents a connected set of words that form a phrase using the regular form of the term. Furthermore, part-of-speech (POS) labeling enhances the consistency of the review by focusing on accuracy and memory. Applying this technique to the lexicons under study can help eliminate irrelevant word clusters or those that may cause issues with document identification.

Snow-ball stemming

The natural language processing method known as stemming groups words with similar meanings under a single stem, reducing word inflections to their simplest forms. In this case, the Snowball Stemmer (SBS), an improved version of the Porter Stemmer often referred to as the Porter2 stemming method[33], is used. Transforming terms into stems is accomplished by probabilistically altering the context of a word's ending sequence. Earlier stemming techniques failed to consider the importance of small character sets within words. Conversely, lemmatization is a technique that reduces words to their base or dictionary form while retaining all of their letters.

Balancing the data using modified conditional generative adversarial network (MCGAN)

Before sending it for the prediction process, the data extracted from Kaggle needs to be balanced. Hence, the MCGAN model is employed in the study. GAN is a deep learning technique that replicates the idea of two-human zero-sum games from game theory and is used to manage complicated, large-scale data in real-time. Oversampling the available data is achieved by balancing minority and majority classes using this technique[34]. Two neural networks, the generator (g) and the discriminator (D), were combined into GAN. To examine the distribution of real input samples, let's utilise g. $S = \{s1, s2, \ldots, sn\}$ and generate fresh data sets, with D serving as a binary classifier to regulate if the data in question is original or generated (z). To undo the weight loss, the categorised result goes back to g and D. D continues to classify original and engendered data samples until the process is complete. To achieve a Nash equilibrium among g and D, the learning method use a mini-max game. This is the representation of the optimisation function for GAN:

$$\min_{g} \quad \max_{D} \; V(g, D) = E_{s\sim p(s)}\left[\left[\log D(S)\right] + E_{z\sim p(z)}\left[\log\left(1 - D\left(g\left(z\right)\right)\right)\right]\right] \tag{1}$$

where D(s) is the probability that case s is real data, p(s) is the distribution of real data instances, and g(z) is the noise info implied to the search space. To differentiate between generated and real data, the D(s) samples also

require more information than D(g(z)). Mode collapse is an issue with the conventional GAN method; rather than considering the full distribution, the results only lead to one class. When the sample distribution is indeed multi-modal, this issue could arise.

The study's suggested conditional generative adversarial network (CGAN) addresses the aforementioned problem by modifying the GAN Model and feeding it real samples, noise information, and category data as inputs into the g and D with loss policy. CGAN learns well when used in conjunction with preexisting distribution samples.

$$\min_{g} \quad \max_{D} \; V(g,D) = E_{s\sim p(s)}\left[\left[\log D(S|x)\right] + E_{z\sim p(z)}\left[\log\left(1 - D(g(z|x))\right)\right]\right] \tag{2}$$

is given in Eq. (1), with x representing the class information and the other parameters. The class disparity issues are typically minimised by using GAN and CGAN to construct a set of instances. Unfortunately, their implementation of the Jensen Shannon distributer leads to the unsatisfactory and unrealistic overlay of real and created case scattering. Since the D is optimally trained, it has the potential to cause the classic collapse and eliminate gradient problems[35].

In this study, to addressed these concerns by modifying CGAN and combining the Lipschitz border with the Wasserstein distance. The D, labels to g are all addressed by these integrated strategies. Here, to use D to find the real and generated instances s. In the event that D is unable to differentiate between real and produced s, to adjust the g and train D, and likewise for D. to keep doing this until to find a value for the loss rate that is around half. By removing the dwindling gradients that impact D during training, the suggested model can generate quantified pattern data, therefore balancing the imbalanced dataset[36]. What follows is an illustration of the MCGAN fitness function.

$$\min_{g} \; V(g,D) = \max_{D} \; \left\{E_{s\sim p(s)}\left[D\left(S|x\right)\right] - E_{z\sim p(z)}\left[D\left(s|x\right)\right] - \varphi E_{s\sim p(\omega)}\left[\nabla_s D(s|x) - 1\right]^2\right\} \tag{3}$$

where ϕ symbolizes an artificial factor, $\|\nabla_s D(s|x)\|$ stipulates the calculating pattern for s in $D(s)$, and $s \sim p(w)$ regulates the centre site facts - on $p(w)$ and $p(g)$.

Feature extraction

Sentiment analysis relies heavily on the feature extraction process. One common use is to transform high-dimensional data into a more manageable low-dimensional characteristic. This is where sentiment analysis gets its start by extracting textual traits. To further assess our strategy, to propose using LFMI, which stands for log-term frequency-based adapted inverse class frequency[37]. The approach also includes testing and training, with training being when textual bits are extracted.

It is common practice to perform the word weighting of the input reviews following the pre-processing operation. In a text including review sentences, the term frequency (T_{fq}) is used to measure how often a word appears. On its own, though, it isn't enough to determine a text's quality; the terms that appear more often have a stronger impact. Supervised term weighting approaches that incorporate class information from reviews are becoming more popular. Consequently, this scenario combines T_{fq} with a supervised term weighting approach. The term appears on training reviews a certain fraction of the time, and this proportion is called the inverse class frequency (I_{fq}). Long term frequency, abbreviated as LT_{fq}, is the output of T_{fq} data after log normalisation. The computation of T_{fq} for each term in the pre-processed dataset is called LTF-centered term weighting. Next, the MICF, a modified of MI_{fq}, is calculated for every word. Because different scores for each term must contribute differently to the aggregate term score, the alteration of MI_{fq} is implemented in this situation. Accordingly, different class-specific scores need different weights, and the final term "score" is the sum of all the class-specific scores, taking into account the weights. Using the aforementioned combination of techniques, Eq. (4) is the proposed method for term weighting.

$$LT_{fq} - MI_{fq}\left(t_m\right) = LT_{fq}\left(t_m\right) * \sum\nolimits_{n=1}^{x} w_{mn}\left[I_{fq}\left(t_m\right)\right] \tag{4}$$

where w_{mn} stands for the specific weighting issue for term t_{m} for class c_{m}, which is well-defined as

$$w_{mn} = log\left(1 + \frac{r_x \overrightarrow{t}}{max\left(1, r_x \overleftarrow{t}\right)} . \frac{r_x \overrightarrow{t}}{max\left(1, r_x \widehat{t}\right)}\right) \tag{5}$$

The weighting factor describes the process by which the available datasets are given relative importance. As count of phrase t_m, $r_x \overrightarrow{t}$ stands for the count of reviews in other classes that utilise the term tm. The amount of r_x in classes other than cm that do not include the term t_m is represented by rxte, while the sum of reviews in include the term is represented by $r_x \overrightarrow{t}$. By utilising the constant 1, negative weights are ignored. The minimal denominator is set to 1 in the event where $r_x \overrightarrow{t}$ =0, in order to avoid problem in the most extreme instance. An innovative term weighting system centred on the MICF is called $LT_{fq} - MI_{fq}\left(t_m\right)$. *The* $LT_{fq}\left(t_m\right)$ *and* $I_{fq}\left(t_m\right)$ formulas may be written as

$$LT_{fq}\left(t_m\right) = \log(1 + T_{fq}\left(t_m, r_x\right)) \tag{6}$$

where $T_{fq}\left(t_m, r_x\right)$ characterizes the total incidence of a phrase t_m occurring $revr_{\mathrm{x}}$.

$$I_{fq}(t_m) = log\left(1 + \frac{N}{C(t_m)}\right) \tag{7}$$

Where $C(t_m)$ is the sum of all the classes in reviews that contain the word tm, and N is the total sum of classes. The features of the dataset are characterized as Fx = F1; F2; ……F3; …:Fp following term weighting, where F1; F2; ……F3; …::Fp is the sum of weighted words from dataset.

Prediction using neural network formulations

The following is an explanation of how the Multi-Layered Attention-Based Encoder-Decoder Temporal Convolutional Neural Network (MLA-EDTCNet), used in this work, makes product predictions.

Encoder–decoder architecture

Neural networks with recurrent layers, gated recurrent units have a solid reputation in machine translation, language modelling, and sequence modelling. Many publications aim to advance recurrent language models that use an encoder-decoder architecture. Our goal in this research was to encode a variable input sequence represented as using an encoder-decoder architecture. $x^{(m)} = \left([p,\ s,\ v]^{\{1\}}, [p,\ s,\ v]^{\{2\}}, \ldots, [p,\ s,\ v]^{\{i\}}, \ldots, [p,\ s,\ v]^{\{I\}}\right)$ into a fixed-length vector. A fully connected layer plus a gated recurrent layer make up the decoder, which will process this information and produce a sequence of displacement vectors y(m) = (d{1}, d{2},…d{i},…, d{I}) of variable length. So, to put it another way, the suggested model is like a brain mechanism that enables mechanical translator that converts the mechanical values given in the x(m) sequence to the y(m) sequence of the plate's midpoint displacement. The next section presents attention the model, after which the limitation of a addressed. In the same way that LSTMs and GRUs can handle sequential data, CNNs can as well. But it can't handle input sequences of varying lengths by design. What this implies is that processing 500 timesteps is the default for a 1D-CNN layer.

If the sequence length is less than 500 timesteps or greater than 500 timesteps, an error message will be provided. To get into more detail, let's say we're using a Finite Element simulation to calculate the displacement behaviour using the NN. For the sake of argument, let's say we're now at time step 100 and wish to assess the answer at time step 101. For the purpose of making displacement behaviour predictions, RNN-based NNs store sequence information in take in both the input at besides the internal variables from the preceding time-step.

GRU based encoder–decoder

Section "Results and discussion" presents the outcomes of the comparison between the computational graphs and the proposed architecture, which demonstrated superior performance, particularly in predictions made on the validation set, where the range of values lies outside the training range. Before proceeding, it is important to emphasize a key aspect of the encoding architecture. The encoder processes an input sequence of I time steps; however, as the sequence length increases, memorization and conversion into a fixed-length representation become more challenging tasks. It may be difficult for the decoder to summarize lengthy input sequences when only the final hidden state of the encoder is provided. Numerical stability requires processing long sequences, which necessitates very short step sizes for explicit dynamic shock wave-loaded plates. To address this issue, a modified attention mechanism is introduced to redirect the model's focus from the entire encoded sequence to the specific information needed to make displacement predictions at time step I.

Attention based encoder–decoder architecture

The implicit evolution of a path-dependent problem can be tracked using variables of a gated recurrent layer. The encoder-decoder architecture's ability to handle and convert mechanical sequences was tested, demonstrating reliable performance even for unknown input sequences, largely due to the attention mechanism addressing translation challenges. To meet the requirements of the prediction tasks, the encoder-decoder architecture underwent the following modifications:

- A learnable dense layer replaced the embedding layer at the edge of the information transfer interface in the decoder.
- Mean squared error replaced probability-based loss functions.

These adjustments are essential as they bridge the gap between translation and prediction tasks in language processing. The proposed approach assigns the task of encoding the source sequence to the encoder, implemented as a GRU layer. Instead of transmitting the encoder's output at the end of the first-time step, a modified attention mechanism is introduced, which performs a soft search over the encoded sequence and utilizes the necessary information to decode an output at the beginning of the process. The following equations, expressed mathematically, implement this soft-searching algorithm:

$$S^{[i]} = tanh\left(W^{s1}o_c^{(m)} + W^{s2}h_1^{(i-1)} + b^s\right) \tag{8}$$

$$a^{[i]} = \left(VS^{[i]} + b^v\right) \tag{9}$$

$$a^{[i]} = \frac{exp\left(a_p^{\{i\}}\right)}{\sum_{k=1}^{l} exp\left(a_p^{\{k\}}\right)} \tag{10}$$

$$C^{[i]} = \sum_{j=1}^{l} \left(a \odot o_e^{(m)}\right) p_j \tag{11}$$

Firstly, a score $S^{[i]}$ among decoder GRU layer $h_1^{(i-1)}$ besides the encoded arrangement $o_e^{(m)}$ is computed following the procedure in[38].

The fully linked layer, as indicated in Eq. (9), is selected as the alignment function and is then given this score. It is the job of this alignment function to figure out how the decoder GRU's output relates to one another. $h_1^{\{i-1\}}$ and the parameters $\left(W^{s1},\ W^{s2}\right) \in R^{n\times n},\ b^s \in R^n, V \in R^{n\times t}$ and $b^v \in R^t$, are optimised by backpropagation in conjunction with the full model to compute a soft alignment, where n is units (cells) besides t = 1 is the features. What comes next is a focus vector $\alpha^{\{i\}}$ is subtracted by Eq. (10) which results in likelihood $\alpha_q^{[i]}\{i\}$ q of the output $o_1^{\{i\}}$ source $x^{\{q\}}$. This using the encoded sequence as input, predicts the output by extracting relevant information $o_1^{\{i\}}$. This eliminates the need for the encoder to condense the entire source sequence into a representation of a predetermined duration and, intuitively, creates an attention mechanism in decoder.

So far, attention has been implemented in decoders in a manner similar to that of machine translation. The next phase involves introducing the change of processing the decoder input using a fully linked layer rather than an embedding layer. One way to put it is as

$$f^{[i]} = W^f y^{\{i-1\}} + b^f \tag{12}$$

with free parameters $W^f \in R^{n\times k}$ and $b^f \in R^n$. This accomplishes the inclusion of care in the decoder. Further, vector $C^{\{i\}}$ besides the vector $f^{\{i\}}$ besides providing as decoder GRU layer. This is characterized by

$$D_1 : \left[f^{[i]}, C^{[i]}\right] \mapsto o_1^{[i]} \tag{13}$$

After the GRU layer's forward pass, operator D_1 computes the output and updates its internal state. It is responsible for executing all the operations included in this pass. $h_1^{[i]} = o_1^{[i]}$. In order to improve the representation's capacity to learn from higher dimensional representation, an additional GRU layer processes the output. The mathematical representation of this transformation is given by the subsequent equation:

$$D_1 : o_1^{[i]} \mapsto o_2^{[i]} \tag{14}$$

Once, the output is computed, the updated $h_2^{[i]} = o_2^{[i]}$ and this is the subsequent step after the dropout layer processing. Regularisation and the network's ability to recognise patterns in data are the goals of the dropout layer. (What follows is a description of how the dropout layer functions). Lastly, the final product $o_2^{[i]} \in R^n$ to the space of the arrangement by means of a completely connected transformation that is articulated as

$$\mathcal{F} : o_2^{[i]} \mapsto \overset{\vee}{y}^{\{i\}} \tag{15}$$

Equation (15) states that the operator F represents the completely connected transformation with hyperbolic tangent as function, besides that output where t = 1, denotes the space of the midpoint displacement sequence before stated. The encoder-decoder architecture's forward pass is now complete. Using this framework, to overcome the bottleneck of poor presentation of sequences $x^{(m)}$ and the findings are detailed in section "Conclusion and future work". The use of soft search is similar to finite element modelling of plates loaded by shock waves. If to are at time step i − 1 and wish to forecast the displacement at step, the attention weights that have been computed indicate that history are given more weight than the other values in the encoded sequence, because to sequence with I time steps sequentially. The progression through models is like an incremental method; in this case, the input at the present from the past are the sole factors that determine the displacement at the subsequent step. Furthermore, the variables in the suggested model $[h^{\{i\}}, h_1^{\{i\}}, h_2^{\{i\}}]$ match the encoder and decoder GRU layers' internal variables that monitor the sequence's progression. They so constitute a decreased nonlinear order reduction of variables in the analysis, as indicated in the preceding sections. Section "Results and discussion" explains the inherited process for training the suggested model. To make the computational graphs data instead of memorising the data itself, a dropout regularisation technique was finally applied.

Temporal convolutional network

To use the TCN from[38] in this study because it has two key features: first, it can map an input sequence of any length to an output of the similar length, and second, the convolution process it uses is causal in nature.

To achieve these two goals, to use a 1D convolution operation in conjunction with the padding technique to make sure the output has the same length as layer. To obtain an output at time step t, the causal requirement is met by convolving exclusively with rudiments from a specific time t steps prior layer. The requirement for network or a high filter size to capture the complete past data is a big drawback of such a system. An exponentially extended receptive field, made possible by a dilated convolution, allows us to circumvent this. Put another way,

given an input sequence and filter f $x \in R^{n}$. When applied to a sequence element e, at time period t, the dilation operation is distinct as

$$H(e) = \sum {}^{s-1}_{i=0} g(i) .x_h \tag{16}$$

in where s is the filter size, i is the system level, and $h = e - d \bullet i$ is path dependence, with $[d = 2]^{i}$ standing for the dilation base in this study. Additionally, for each layer's receptive field, to have $r = (s - 1) \bullet d$; for the full residual block, to have the subsequent reckoning.

$$r_j = 1 + 2.(s-1) .N_b. \left(\sum {}_i d_i \right) \tag{17}$$

where, N_b represents the sum of residual blocks. Here to can see exactly what to do to make the layer more receptive: either use deeper networks or use larger filter size s.

Stabilising the initialised model becomes increasingly crucial as the network depth increases. A residual block is utilised in this investigation. It streamlines training by introducing a skip connection, which gives gradients another way to flow across the computational network. There is a single residual block (Nb = 1) and thirty-two filters (s = 3) that make up the TCN layer. The reason for this is that sequences with a lot of time steps should have more leftover blocks. Using a larger number of dilations d = [1, 2, 4, 8, 16, 32, 64], to deepen the model. You may calculate the model's receptive field using Eq. (17) and this will guarantee that it is greater than 500 timesteps.

Fine-tuning using proposed Ocotillo optimization algorithm (OcOA)

The study employs the Ocotillo Optimization Algorithm (OcOA) [39], which conceptually addresses fundamental challenges in metaheuristic optimization, to fine-tune the parameters of the proposed model. The main objective of OcOA is to balance two conflicting goals: efficiently exploring the solution space and effectively exploiting promising areas. Excessive exploration can waste computational resources, while excessive exploitation may lead to premature convergence on suboptimal solutions. OcOA dynamically adapts its behavior to the problem landscape, offering a more responsive and flexible search process.

The algorithm operates in an iterative cycle: expansive exploration probes a broad area of the solution space, and focused exploitation refines promising regions to identify optimal solutions. By iteratively balancing these phases, the algorithm efficiently converges on the global optimum while avoiding local optima. This balance is achieved by adjusting a set of parameters that govern the adaptive switching between exploration and exploitation, ensuring a well-controlled and efficient search process throughout execution.

Additionally, OcOA incorporates a feedback system that enables it to learn from past iterations. This feedback improves both search direction and solution accuracy. As the algorithm progresses, it becomes more focused on promising solutions and less exploratory, thereby accelerating convergence and improving solution quality. Due to its dynamic adaptability, OcOA is particularly well-suited for solving complex, multimodal, and high-dimensional optimization problems. By leveraging its intelligent and flexible search behavior, OcOA outperforms conventional metaheuristics that struggle to maintain a proper balance between exploration and exploitation.

Experimental setup

In this section, the essential variables, equations, and constants that constitute the Ocotillo Optimization Algorithm (OcOA) are described. The algorithm's adaptive processes, which rely on feedback and stochastic components, are designed to strike a balance between exploration and exploitation. This section outlines the mathematical models used in the algorithm's exploitation stages, as well as the components that make it adaptive based on its past performance.

Constants and parameters The behavior of OcOA is governed by several constants and parameters that influence the algorithm's ability to effectively explore the solution space while refining promising solutions. These key parameters include:

- r1,r2,r3: Random variables uniformly distributed in the range [0,1][0, 1][0,1], used to introduce randomness into the search process and prevent the algorithm from becoming trapped in local optima.
- A: The amplitude of the search step, which dynamically adjusts based on the current position in the solution space. It determines the step size during both exploration and exploitation.
- D: The distance factor, which controls how widely the search process spreads across the solution space. This is critical during the exploration phase to ensure diversity.
- L: The learning factor, which adjusts the search direction based on feedback from previous iterations. This enables the algorithm to incorporate knowledge gained from prior steps.

These constants are essential for maintaining an effective balance between exploration (broad search) and exploitation (local refinement) as the algorithm iterates through the solution space.

Exploration phase The exploration phase in OcOA ensures that the algorithm explores a wide area of the search space, preventing local optima. The position update equation during this phase is given by:

$$\Theta\ (t+1) = \Theta_{\,t} + (A.D) + (r_1.L) \tag{18}$$

- $\Theta\ (t+1)$ is the updated position at the next iteration,

- A · D represents the amplitude and distance of the exploratory step,
- r1 · L introduces stochasticity and learning from prior iterations.

The distance factor D, which ensures diversity in the exploration process, is calculated as follows:

$$D = 2.\frac{\cos\left(\Theta\right)}{(Train)^2} \tag{19}$$

This equation adjusts the search step based on the cosine of the current position Θ, and the factor Train controls the spread of the search. This prevents premature convergence besides allows the algorithm to cover a larger area of the key space.

The amplitude A, which modulates the step size based on the current position Θ, is given by:

$$A = sin\left(\frac{a}{b}\Theta\right) \tag{20}$$

Where a and b are constants. This equation allows the algorithm to dynamically adjust the step size, ensuring that it takes larger steps when necessary and refines the search when appropriate.

To introduce further variation during the exploration phase, the algorithm employs the following mutation equation:

$$O\left(t\right) = \left(r_1.A\right) + \left(\left(r_1.r_2\right).\frac{A.L}{D}\right).\left(r_1 + \left(r_1.r_2\right)A + \left(r_1, r_2.r_3\right)L\right) \tag{21}$$

This mutation equation adds randomness to the position updates by perturbing the current position with stochastic components r1, r2, and r3. The introduction of random factors ensures that the local minima, maintaining a broad search across the solution space.

Exploitation phase Once promising areas of the search space have been exploration, the algorithm transitions to phase. The goal here is to refine the solutions and improve accuracy. The position update equation during exploitation is as follows:

$$\Theta\left(\mathrm{t}+1\right) = \mathrm{Gaussion}\left(\mu\right) + L_1.r_1.\frac{A}{iTrain} + \frac{F.A}{o} \tag{22}$$

Where:

- Gaussian(μ) introduces a Gaussian mutation to prevent stagnation,
- $L_1.r_1.\frac{A}{iTrain}$ rain refines the solution based on previous knowledge and the amplitude A,
- $\frac{F.A}{o}$ modulates the intensity of exploitation using the function F and the normalization factor o.

The function F, which regulates the intensity of exploitation, is calculated as:

$$F = \sum{}^{n=10}_{i=0}\left(\frac{\sin n\Theta}{(Train)^2}\right) \tag{23}$$

This function sums the influence of previous iterations to focus on promising areas of search space, guiding the algorithm toward regions that are more likely to contain the global optimum.

An alternative form of the exploitation equation introduces further refinement:

$$\Theta\ \left(\mathrm{t}+1\right) = \left(r_2.i_2\right).\left(\frac{K.A}{\left(i\,Train\right)^2}\right)^2 \tag{24}$$

This equation incorporates the convergence factor K, which adjusts the step size as the algorithm hones in on the optimal solution.

Feedback handling Feedback plays a critical role in the OcOA by allowing the algorithm to learn from previous iterations and adjust its search behavior accordingly. The learning factor L, which influences the search direction based on feedback, is defined as:

$$L = 1 - \frac{2sin\Theta}{\left(1 + 8\,cos\Theta\,\right)^2} \tag{25}$$

This equation adjusts the search direction by incorporating the experience from previous iterations, ensuring that the algorithm becomes more focused over time and improves its search efficiency.

Additionally, the convergence factor K, which modulates the exploitation process, is calculated as:

$$K = 1 - \left(\frac{Gaussian\,(\mu\,)}{A + D} \right)^2 \tag{26}$$

This factor ensures that the algorithm gradually reduces its exploration and focuses more on exploitation as it approaches convergence, preventing the search from oscillating unnecessarily between wide search and local refinement. By dynamically adjusting the balance between exploration and exploitation and incorporating feedback mechanisms, the OcOA ensures efficient and robust performance across a variety of complex optimization problems.

Product recommendation system

Product recommendation systems are a type of filtering system that considers users' preferences, past activities, and other relevant data to provide product suggestions. These systems are commonly used in online marketplaces, video streaming services, and e-commerce platforms to enhance user experience and boost revenues. The proposed product recommendation system integrates the SA model with collaborative filtering (CF). Using CF, a system was developed to recommend items.

User-based CF identifies similar users by analyzing their past interactions with products. By finding users who have reviewed or purchased items similar to those being considered, the system can make personalized recommendations. For instance, products preferred by similar users can be recommended to a target user, even if the target user has not yet interacted with those products. Item-based CF, on the other hand, locates items with similar attributes based on past user interactions. It searches for items that users have rated or purchased together to recommend similar products to the target user. The underlying idea is that if a user likes a particular item, they are more likely to be interested in related items. To further improve the accuracy of predictions, the SA model was combined with the most effective recommendation system.

One popular method, known as collaborative filtering (CF), analyzes the habits and preferences of a group of users to provide tailored recommendations. It assumes that users will likely maintain their original preferences and tastes over time. By analyzing past interactions between users and items, the CF approach can identify commonalities and patterns. The CF method does not require prior knowledge of the characteristics or attributes of items and users. Instead, it relies solely on observable interactions between users and items. This is particularly useful when user preferences evolve over time or when information about item quality is limited.

The user-item matrix is constructed using the collected data. In this matrix, items are represented by columns and users by rows, showing the interactions between users and items. Elements in the matrix, such as ratings or binary values indicating interaction status, represent the relationship between users and items. The next step is to analyze the user-item matrix and determine the similarity between users or items.

In this case, the cosine similarity metric is used to measure the degree of similarity. Cosine similarity calculates the cosine of the angle between two vectors. The resulting value ranges from − 1 to 1, where 1 indicates identical vectors, 0 indicates no similarity, and − 1 indicates polar opposites. The formula for cosine similarity is as follows:

$$Cosine\ Similarity\ \ (M, N)\ = \frac{(M\ \cdot\ N)}{(|M|\ *\ |N|)} \tag{27}$$

The dot product of vectors $M\ and\ N$ is denoted by $M\ \bullet\ N$, and the Euclidean (magnitudes) of vectors M besides N are represented by ||M|| and ||N||, respectively.

In user-based collaborative filtering, similarity computation is based on how users engage with and prefer various items. Analyzing the activity of users who exhibit similar interaction patterns is one approach to offering recommendations. Item-based collaborative filtering, on the other hand, considers user engagement with items to identify comparable products. Recommendations are then generated by finding similar items based on the frequency with which the same users interact with them.

The first step in the neighborhood selection process is determining the degree of similarity between two users or items. The purpose of this stage is to select a group of similar users or items to generate recommendations. However, a larger neighborhood can significantly increase computational complexity and potentially impact recommendation quality. Once the neighborhood is selected, recommendations are generated based on user interests or related products.

More than 25 products were considered, and the best recommendation method was selected based on user ratings, identifying the items most likely to be purchased. Finally, to further enhance the recommendations, the SA model was integrated with the recommendation system.

Results and discussion

To conduct the research, a system with an Intel Core i5-7200 processor and 8 GB of internal memory was utilized. The processor operates at a clock speed of 2.7 GHz. A dedicated user interface (UI) and the Jupyter Notebook environment were used to perform the procedures on Windows 10, a 64-bit operating system.

Validation analysis of proposed model

The experiments are conducted based on three emotions: positive, negative, and neutral, using various metrics, as shown in Figs. 2, 3, 4 and 5. The existing works referenced in section "Related work" utilize different review datasets; therefore, this study considered our dataset and tested it with a basic implementation, with the results averaged.

A graphical comparison of the proposed model demonstrates its superior performance, achieving an accuracy of 83.6, an F1-score of 0.788, a sensitivity of 0.841, and an AUC of 0.858. In comparison, the LSTM

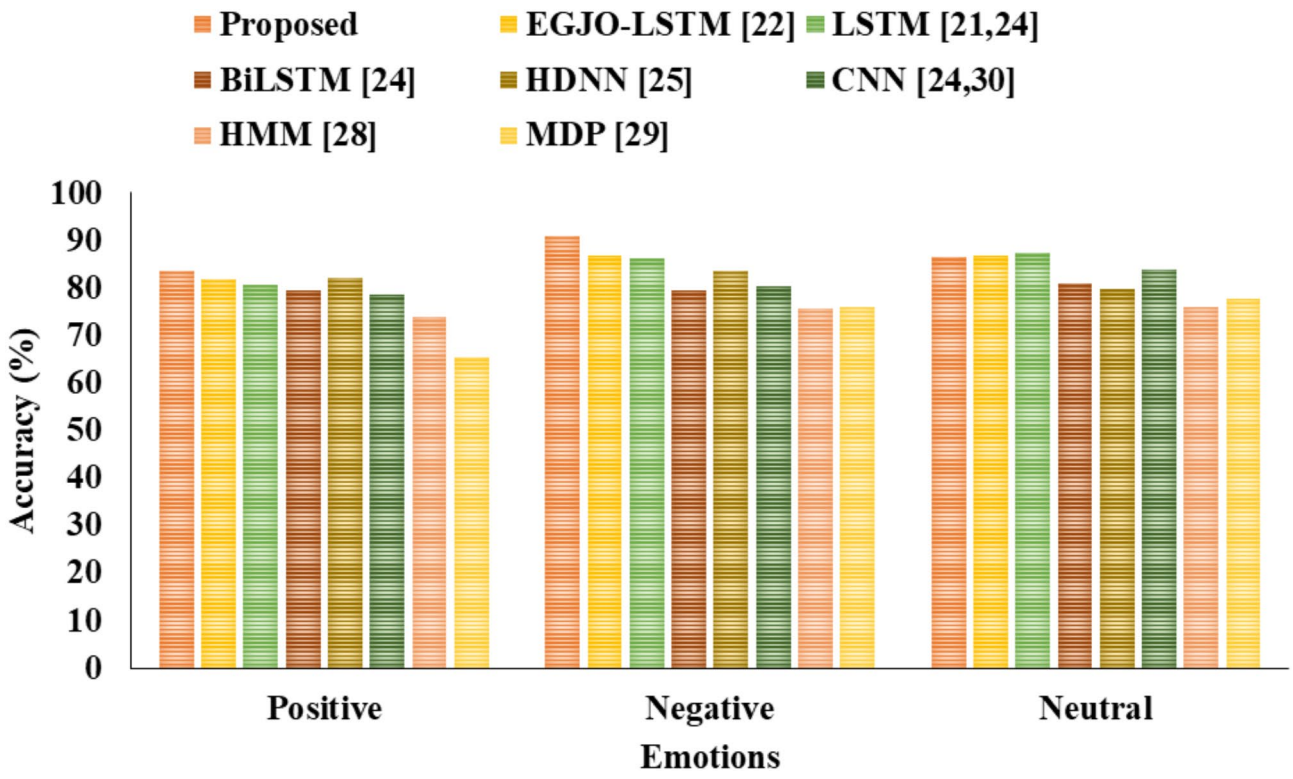

Fig. 2. Graphical comparison of proposed model.

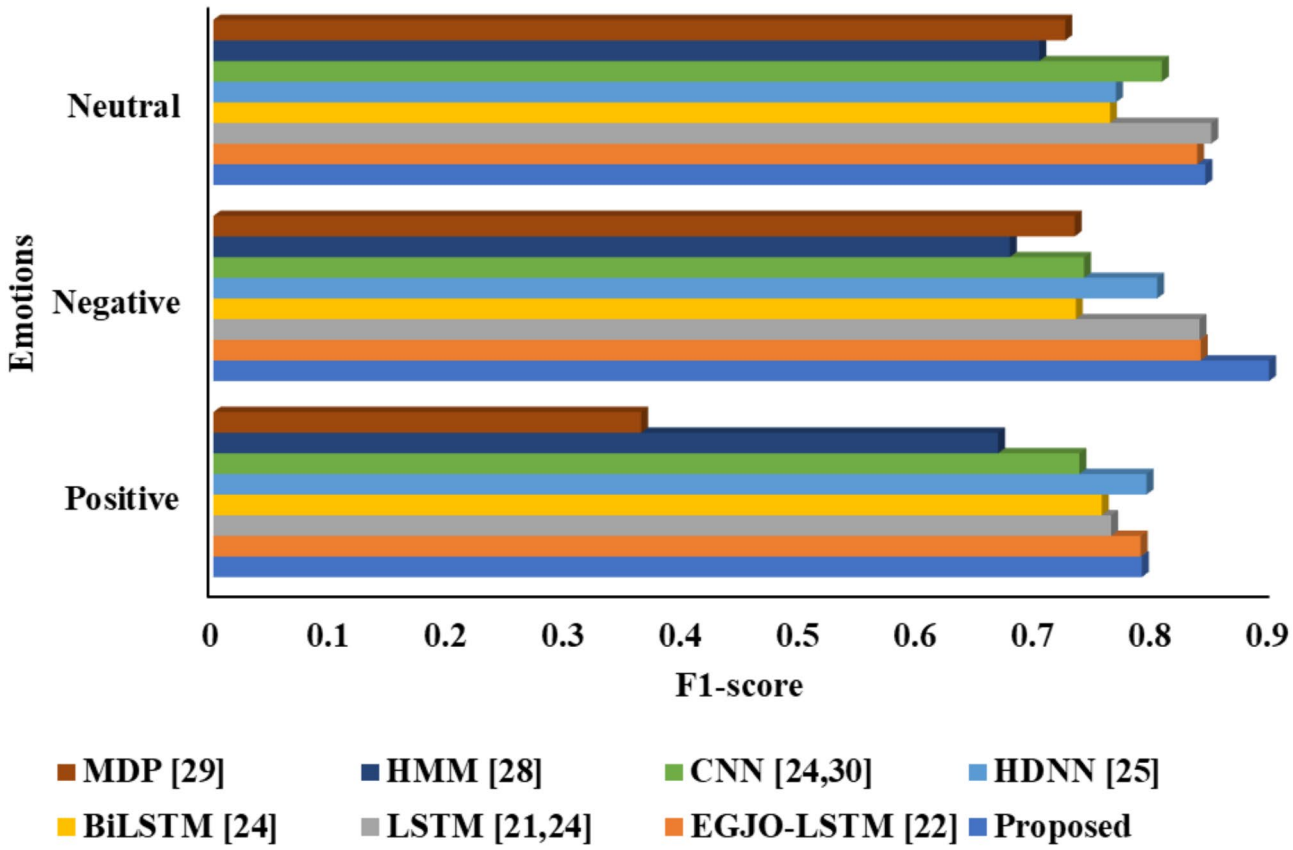

Fig. 3. Visual Analysis of different models.

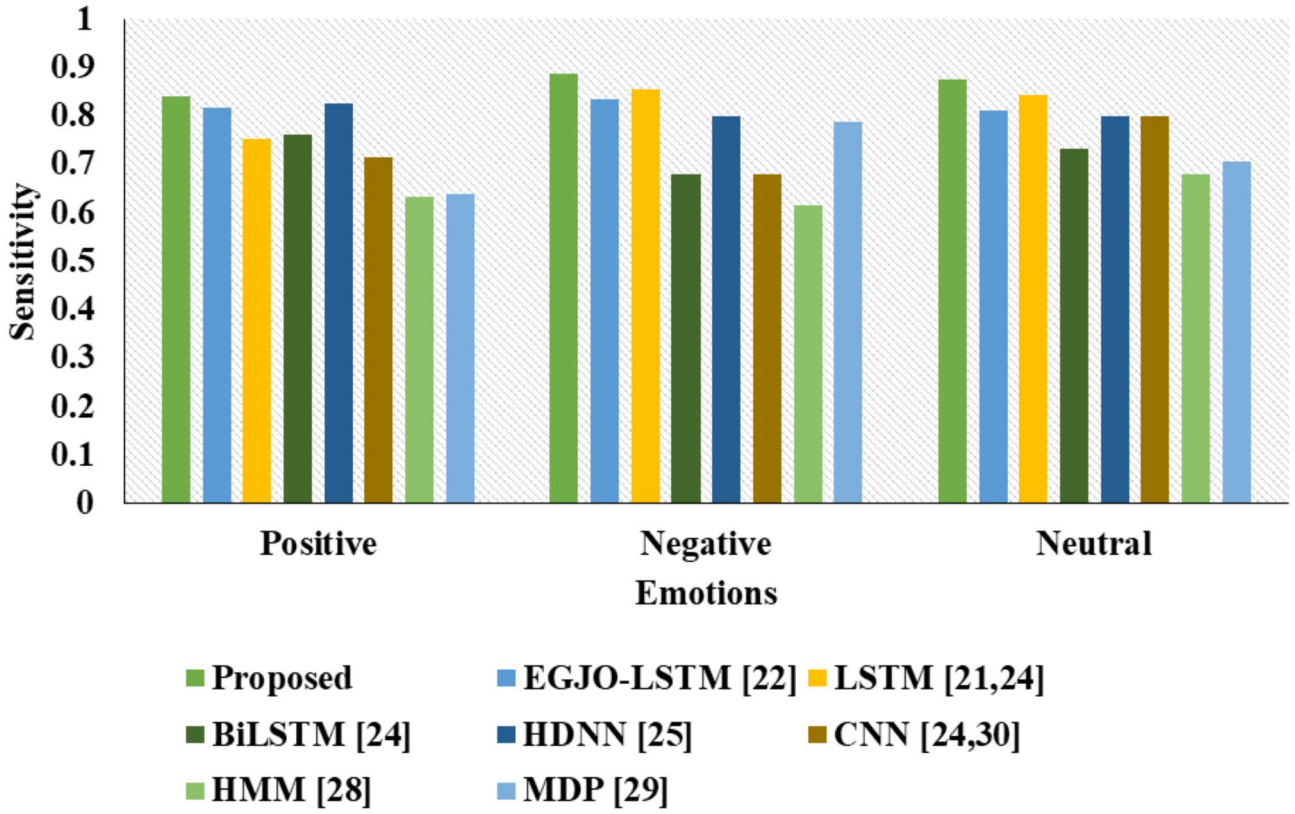

Fig. 4. Description of projected model.

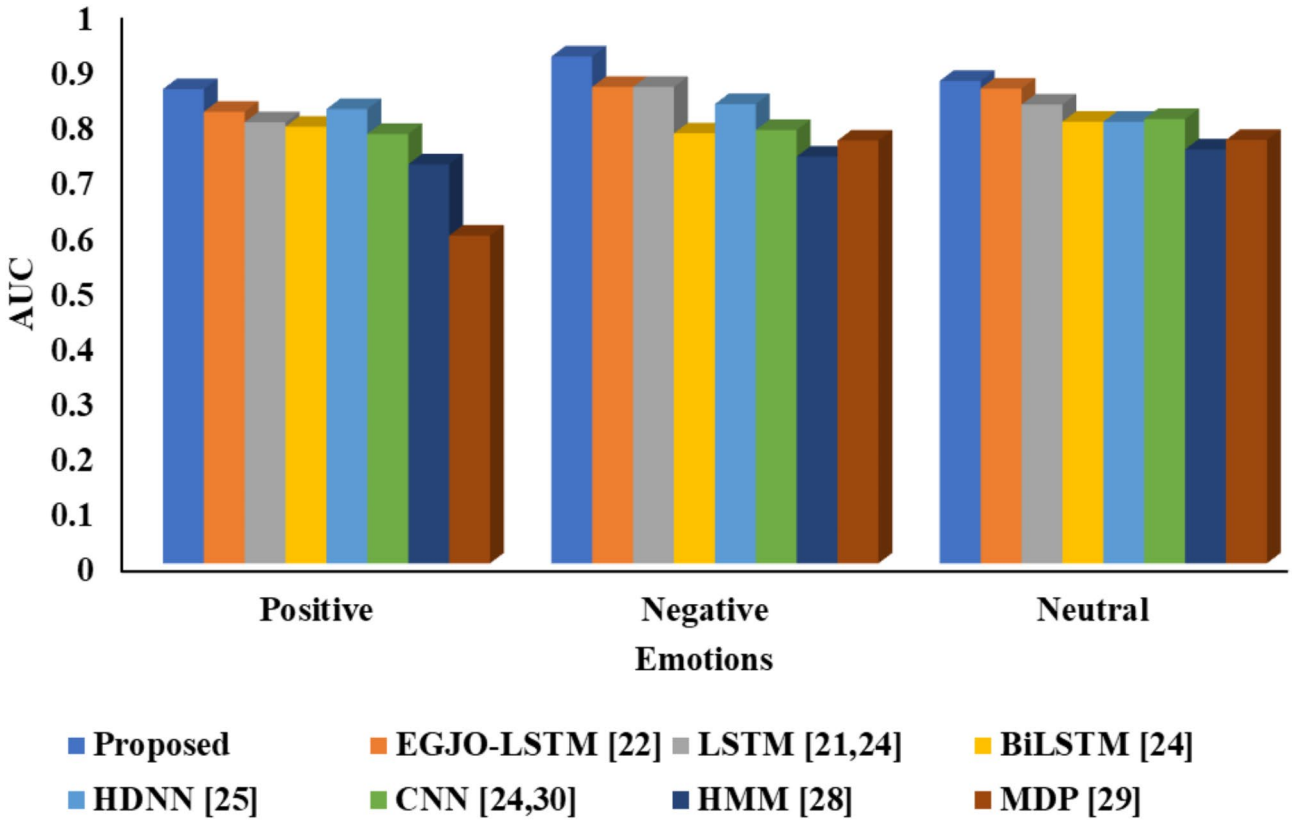

Fig. 5. Study of proposed model with existing procedures.

model[21,24] achieved an accuracy of 80.6, an F1-score of 0.762, a sensitivity of 0.752, and an AUC of 0.798. The BiLSTM model[24] reached an accuracy of 79.5, an F1-score of 0.754, a sensitivity of 0.761, and an AUC of 0.790. The HDNN model[25] recorded an accuracy of 82.1, an F1-score of 0.792, a sensitivity of 0.825, and an AUC of 0.822. The CNN model[24,30] achieved an accuracy of 78.7, an F1-score of 0.735, a sensitivity of 0.715, and an AUC of 0.777. The HMM model[28] showed an accuracy of 73.8, an F1-score of 0.666, a sensitivity of 0.633, and an AUC of 0.722. Finally, the MDP model[29] performed the lowest, with an accuracy of 65.5, an F1-score of 0.363, a sensitivity of 0.638, and an AUC of 0.593. This comparison highlights the effectiveness of the proposed model over other existing methods.

The proposed model achieved an accuracy of 90.9%, an F1-score of 0.896, a sensitivity of 0.886, and an AUC of 0.917. In comparison, the EGJO-LSTM[22] achieved an accuracy of 86.7%, an F1-score of 0.838, a sensitivity of 0.834, and an AUC of 0.862. The LSTM[21,24] recorded an accuracy of 86.3%, an F1-score of 0.837, a sensitivity of 0.853, and an AUC of 0.862. The BiLSTM[24] reached an accuracy of 79.5%, an F1-score of 0.732, a sensitivity of 0.678, and an AUC of 0.778. The HDNN[25] achieved an accuracy of 83.7%, an F1-score of 0.801, a sensitivity of 0.798, and an AUC of 0.831. The CNN[24,30] recorded an accuracy of 80.3%, an F1-score of 0.739, a sensitivity of 0.678, and an AUC of 0.784. The HMM[28] achieved an accuracy of 75.7%, an F1-score of 0.676, a sensitivity of 0.614, and an AUC of 0.736. Finally, the MDP[29] accomplished an accuracy of 76.1%, an F1-score of 0.731, a sensitivity of 0.788, and an AUC of 0.765.

The proposed model achieved an accuracy of 86.5%, an F1-score of 0.842, a sensitivity of 0.874, and an AUC of 0.873. In comparison, the EGJO-LSTM[22] achieved an accuracy of 86.7%, an F1-score of 0.835, a sensitivity of 0.810, and an AUC of 0.859. The LSTM[21,24] reached an accuracy of 87.5%, an F1-score of 0.847, a sensitivity of 0.844, and an AUC of 0.830. The BiLSTM[24] achieved an accuracy of 81%, an F1-score of 0.761, a sensitivity of 0.733, and an AUC of 0.799. The HDNN[25] recorded an accuracy of 79.9%, an F1-score of 0.766, a sensitivity of 0.798, and an AUC of 0.799. The CNN[24,30] achieved an accuracy of 84%, an F1-score of 0.805, a sensitivity of 0.798, and an AUC of 0.804. The HMM[28] achieved an accuracy of 76.1%, an F1-score of 0.701, a sensitivity of 0.678, and an AUC of 0.749. Finally, the MDP[29] achieved an accuracy of 77.6%, an F1-score of 0.723, a sensitivity of 0.706, and an AUC of 0.766.

Discussion

The results demonstrate the superior performance of the proposed model compared to existing methods across key evaluation metrics, including accuracy, F1-score, sensitivity, and AUC. The proposed MLA-EDTCNet model enhances interpretability by making sentiment-driven recommendations more transparent. Unlike traditional recommendation models, our approach explicitly identifies key sentiment-influencing features in user reviews, allowing e-commerce platforms to provide explainable recommendations where users can see how their past reviews and sentiment patterns influence product suggestions. Retailers can analyze customer sentiment data to optimize product placement and inventory decisions, ensuring that high-demand products with positive sentiment are prioritized. Additionally, marketing teams can leverage sentiment trends to personalize advertising and promotions based on real-time consumer feedback, enhancing customer engagement and driving sales. The integration of sentiment analysis (SA) with collaborative filtering (CF) effectively enhances recommendation precision by incorporating user sentiment into the decision-making process. This approach addresses limitations observed in traditional models, such as the inability to balance exploration and exploitation effectively or handle diverse user preferences dynamically.

The proposed model consistently outperformed baseline models such as LSTM, BiLSTM, and HDNN, showcasing the robustness of the Multi-Layered Attention-Based Encoder-Decoder Temporal Convolutional Neural Network (MLA-EDTCNet) and its ability to capture complex user-item interactions. The Ocotillo Optimization Algorithm (OcOA) further contributed by fine-tuning model parameters, ensuring balanced exploration and exploitation during training, and enhancing predictive capabilities.

Compared to state-of-the-art methods like EGJO-LSTM, the proposed model achieved notable improvements, particularly in sensitivity and AUC, which are critical for personalized recommendations. These results highlight the significance of combining advanced optimization techniques and sentiment-driven approaches to address challenges such as class imbalance, limited scalability, and evolving user preferences in recommendation systems. Future work could explore scalability and adaptability in real-time environments.

Limitations

- **Scalability**: While the proposed model demonstrates robust performance on the tested dataset, its scalability to extremely large and dynamic datasets in real-time environments remains unexplored.
- **Dependency on Sentiment Analysis**: The model's reliance on accurate sentiment analysis (SA) could lead to reduced performance if the SA component misclassifies user sentiments, particularly for ambiguous or context-specific reviews.
- **Computational Overhead**: The integration of advanced techniques such as the Ocotillo Optimization Algorithm (OcOA) and the MLA-EDTCNet increases computational complexity, which may limit its applicability in low-resource environments.
- **Generalization to Diverse Domains**: The model has been validated using specific datasets (e.g., Amazon reviews), but its effectiveness in diverse application domains or non-English datasets has not been fully established.

Conclusion and future work

This research proposed a robust product recommendation system that integrates sentiment analysis (SA) with collaborative filtering (CF) to enhance recommendation accuracy and user satisfaction. By employing a Multi-

Layered Attention-Based Encoder-Decoder Temporal Convolutional Neural Network (MLA-EDTCNet) and fine-tuning it with the Ocotillo Optimization Algorithm (OcOA), the model effectively balances exploration and exploitation, addressing challenges such as class imbalance and evolving user preferences. Experimental results demonstrate the proposed model's superiority over existing methods, achieving higher accuracy, F1-score, sensitivity, and AUC. The integration of SA into the recommendation framework ensures personalized and relevant product suggestions, significantly improving the user experience.

Despite its advantages, the model has limitations, including scalability challenges, computational overhead, and generalizability to diverse domains. Future research will focus on enhancing the system's scalability to handle large, real-time datasets and optimizing computational efficiency to make the model suitable for resource-constrained environments. Additionally, efforts will be made to adapt the system for multilingual and domain-diverse datasets, extending its applicability to broader contexts. Exploring explainability in recommendations and integrating hybrid approaches for dynamic user modeling will also be prioritized to further refine the system's effectiveness.

Data availability

The datasets used and/or analyzed during the current study available from the corresponding author on reasonable request.

Received: 3 January 2025; Accepted: 19 February 2025
Published online: 25 February 2025

References

1. Solairaj, A., Sugitha, G. & Kavitha, G. Enhanced Elman Spike neural network based sentiment analysis of online product recommendation. *Appl. Soft Comput.* **132**, 109789 (2023).
2. Wang, E., Yang, Y., Wu, J., Liu, W. & Wang, X. An efficient prediction-based user recruitment for mobile crowdsensing. *IEEE Trans. Mobile Comput.* **17**(1), 16–28. https://doi.org/10.1109/TMC.2017.2702613 (2018).
3. Ramshankar, N. & Joe Prathap, P. M. Reviewer reliability and XGboost Whale optimized sentiment analysis for online product recommendation. *J. Intell. Fuzzy Syst.* **44**(1), 1547–1562 (2023).
4. Zuo, C., Zhang, X., Yan, L. & Zhang, Z. GUGEN: global user graph enhanced network for next POI recommendation. *IEEE Trans. Mobile Comput.* **23**(12), 14975–14986. https://doi.org/10.1109/TMC.2024.3455107 (2024).
5. Song, W. et al. TalkingStyle: personalized speech-driven 3D facial animation with style preservation. In *IEEE Transactions on Visualization and Computer Graphics*. https://doi.org/10.1109/TVCG.2024.3409568 (2024).
6. Pughazendi, N., Rajaraman, P. V. & Mohammed, M. H. Graph sample and aggregate attention network optimized with barnacles mating algorithm based sentiment analysis for online product recommendation. *Appl. Soft Comput.* **145**, 110532 (2023).
7. Lianwei Wu, Y., Gao, L. C., Wang, Z. & Zhang, Y. Multimodal fusion and inconsistency reasoning for explainable fake news detection. *Inform. Fusion* **100**, 101944 (2023).
8. Elahi, M., Kholgh, D. K., Kiarostami, M. S., Oussalah, M. & Saghari, S. Hybrid recommendation by incorporating the sentiment of product reviews. *Inf. Sci.* **625**, 738–756 (2023).
9. Song, L., Chen, S., Meng, Z., Sun, M. & Shang, X. A Fine-Grained multimodal sentiment analysis dataset based on stock comment videos. *IEEE Trans. Multimedia* **26**, 7294–7306. https://doi.org/10.1109/TMM.2024.3363641 (2024).
10. Elzeheiry, S., Gab-Allah, W. A., Mekky, N. & Elmogy, M. Sentiment analysis for e-commerce product reviews: current trends and future directions (2023).
11. Sharma, A. K. et al. An efficient approach of product recommendation system using NLP technique. *Mater. Today: Proc.* **80**, 3730–3743 (2023).
12. Zhan, Z. & Xu, B. Analyzing review sentiments and product images by parallel deep Nets for personalized recommendation. *Inf. Process. Manag.* **60**(1), 103166 (2023).
13. Yang, K. How to prevent deception: A study of digital deception in visual poverty livestream. *New. Media Soc.* https://doi.org/10.1177/14614448241285443 (2024).
14. Karabila, I., Darraz, N., El-Ansari, A., Alami, N., Mallahi, E. & M Enhancing collaborative filtering-based recommender system using sentiment analysis. *Future Internet* **15**(7), 235 (2023).
15. Kumar, S., Pandey, A. & Shrivastava, D. V. A brief overview on various aspects of recommendation system based on sentiment analysis. Available at SSRN 4602378 (2023).
16. Peng, Y., Zhao, Y. & Dong, J. Jiangping Hu, Adaptive opinion dynamics over community networks when agents cannot express opinions freely. *Neurocomputing* **618**, 129123 (2025).
17. Ding, J. et al. DialogueINAB: an interaction neural network based on attitudes and behaviors of interlocutors for dialogue emotion recognition. *J. Supercomput.* **79**, 20481–20514. https://doi.org/10.1007/s11227-023-05439-1 (2023).
18. Periakaruppan, S., Shanmugapriya, N. & Sivan, R. Self-attention generative adversarial capsule network optimized with atomic orbital search algorithm based sentiment analysis for online product recommendation. *J. Intell. Fuzzy Syst.* **44**(6), 9347–9362 (2023).
19. Pan, F., Li, Y. & Hu, B. Research on sentiment analysis and personalized recommendation based on agricultural product reviews. In *2023 9th International Conference on Computer and Communications (ICCC)* 2438–2442 (IEEE, 2023).
20. Liu, Z., Liao, H., Li, M., Yang, Q. & Meng, F. A deep learning-based sentiment analysis approach for online product ranking with probabilistic linguistic term sets. *IEEE Trans. Eng. Manage.* **71**, 6677–6694 (2023).
21. Thomas, R. & Jeba, J. R. A novel framework for an intelligent deep learning based product recommendation system using sentiment analysis (SA). *Automatika* **65**(2), 410–424 (2024).
22. Rasappan, P., Premkumar, M., Sinha, G. & Chandrasekaran, K. Transforming sentiment analysis for e-commerce product reviews: hybrid deep learning model with an innovative term weighting and feature selection. *Inf. Process. Manag.* **61**(3), 103654 (2024).
23. Qin, J., Zeng, M., Wei, X. & Pedrycz, W. Ranking products through online reviews: a novel data-driven method based on interval type-2 fuzzy sets and sentiment analysis. *J. Oper. Res. Soc.* **75**(5), 860–873 (2024).
24. Wasi, A. A., Fahim, E. H., Inova, N. T., Fahim, A. A. & Preeti, T. T. Hybrid recommendation system of intelligent captioning using deep learning networks (Doctoral dissertation, Brac University) (2024).
25. Dey, R. K. & Das, A. K. Neighbour adjusted dispersive flies optimization based deep hybrid sentiment analysis framework. *Multimedia Tools Appl.* **2024**, 1–24 (2024).
26. Goularte, F. B., da Graça Martins, B. E., da Fonseca Carvalho, P. C. Q. & Won, M. SentPT: a customized solution for multi-genre sentiment analysis of Portuguese-language texts. *Expert Syst. Appl.* **245**, 123075 (2024).
27. Danyal, M. M. et al. Proposing sentiment analysis model based on BERT and XLNet for movie reviews. *Multimedia Tools Appl.* **2024**, 1–25. (2024).

28. Wu, Z., Wang, X., Huang, S., Yang, H. & Ma, D. Research on prediction recommendation system based on improved Markov model. *Adv. Comput. Signals Syst.* **8**(5), 87–97 (2024).
29. Iftikhar, A. et al. A reinforcement learning recommender system using bi-clustering and Markov decision process. *Expert Syst. Appl.* **237**, 121541 (2024).
30. Sun, B. et al. A user review data-driven supplier ranking model using aspect-based sentiment analysis and fuzzy theory. *Eng. Appl. Artif. Intell.* **127**, 107224 (2024).
31. Nazari, A., Kordabadi, M. & Mansoorizadeh, M. Scalable and data-independent multi-agent recommender system using social networks analysis. *Int. J. Inform. Technol. Decis. Mak.* **23**(02), 741–762 (2024).
32. https://www.kaggle.com/datasets/lokeshparab/amazon-products-dataset (2024).
33. Deng, S. et al. Learning to compose diversified prompts for image emotion classification. *Comp. Visual Media* **10**, 1169–1183. https://doi.org/10.1007/s41095-023-0389-6 (2024).
34. Zhang, G. et al. Network intrusion detection based on conditional Wasserstein generative adversarial network and cost-sensitive stacked autoencoder. *IEEE Access.* **8**, 190431–190447 (2020).
35. Li, T., Li, Y., Zhang, M., Tarkoma, S. & Pan Hui You are how you use apps: user profiling based on Spatiotemporal app usage behavior. *ACM Trans. Intell. Syst. Technol.* **14**, 4 (2023).
36. Babu, K. S. & Rao, Y. N. MCGAN: modified conditional generative adversarial network (MCGAN) for class imbalance problems in network intrusion detection system. *Appl. Sci.* **13**(4), 2576 (2023).
37. Li, T., Li, Y., Xia, T. & Hui, P. Finding Spatiotemporal patterns of mobile application usage. *IEEE Trans. Netw. Sci. Eng.* https://doi.org/10.1109/TNSE.2021.3131194 (2024).
38. Tandale, S. B. & Stoffel, M. Recurrent and convolutional neural networks in structural dynamics: a modified attention steered encoder–decoder architecture versus LSTM versus GRU versus TCN topologies to predict the response of shock wave-loaded plates. *Comput. Mech.* **72**(4), 765–786 (2023).

Author contributions
All authors contributed equally.

Funding
The authors declare that no funds, grants, or other support were received during the preparation of this manuscript.

Competing interests
The authors declare no competing interests.

Ethics approval
The submitted work is original and has not been published elsewhere in any form or language.

Additional information
Correspondence and requests for materials should be addressed to B.N.J.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.