Suggestion and invention of recipes using bi-directional LSTMs-based frameworks

Sai Nikhil Rao Gona¹ · Himamsu Marellapudi²

Received: 25 November 2020 / Accepted: 31 March 2021

Published online: 13 April 2021

The Author(s) 2021

Abstract

Choosing which recipe to eat and which recipe to avoid isn't that simple for anyone. It takes strenuous efforts and a lot of time for people to calculate the number of calories and P.H level of the dish. In this paper, we propose an ensemble neural network architecture that suggests recipes based on the taste of the person, P.H level and calorie content of the recipes. We also propose a bi-directional LSTMs-based variational autoencoder for generating new recipes. We have ensembled three bi-directional LSTM-based recurrent neural networks which can classify the recipes based on the taste of the person, P.H level of the recipe and calorie content of the recipe. The proposed model also predicts the taste ratings of the recipes for which we proposed a custom loss function which gave better results than the standard loss functions and the model also predicts the calorie content of the recipes. The bi-directional LSTMs-based variational autoencoder after being trained with the recipes which are fit for the person generates new recipes from the existing recipes. After training and testing the recurrent neural networks and the variational autoencoder, we have tested the model with 20 new recipes and got overwhelming results in the experimentation, the variational autoencoders generated a couple of new recipes, which are healthy to the specific person and will be liked by the specific person.

Keywords Bi-directional LSTMs-based ensemble neural networks · Taste, acidity and calorie prediction · Bi-directional LSTMs-based variational autoencoders for generating the new recipe · Bi-directional LSTMs which classifies the recipes

1 Introduction

Good food is one of the most important reasons behind our healthy lives, so food is something that we need to take a lot of care about. There are so many types of dishes in this world among which Indian dishes hold an incredibly special place. Most of the Indian dishes are curries; there are almost 100 to 150 vegetable curry recipes that are popular in all parts of India. The food that is found in India is of high medicinal value, it contains good amounts of protein and vitamins [1]. In India, there are around 30 to 50 different types of ingredients that are added together

in certain amounts to get a good delectable and healthy recipe, so it's really important to add those ingredients in specific amounts to get the best dish out of those ingredients which the person should like eating and should be good for the person's health as well. In India, most people suffer from acidity or acid reflux [2] and this restricts them from eating whatever they like, so the recipe that a person suffering from acidity is eating should be basic and it should be a low calorie one if the person wants to lose weight. New algorithms are being designed these days to integrate artificial intelligence with food. A team of researchers from MIT and various other researchers have

✉ Sai Nikhil Rao Gona, gonasainikhil@gmail.com; Himamsu Marellapudi, himamsu12@gmail.com | ¹Electrical and Electronics Engineering, Chaitanya Bharathi Institute of Technology, Hyderabad, India. ²Electrical and Communication Engineering, Chaitanya Bharathi Institute of Technology, Hyderabad, India.

designed algorithms to predict the ingredients of a recipe by using a photograph of that recipe [3, 4], now we need to know if that recipe is healthy for a specific person with a given health issue or not and if he is going to like the recipe's taste or not, to make this happen the best way is to use artificial neural networks to predict the taste and healthiness. A model was proposed to predict the beer taste quality using artificial neural networks which gave us the knowledge to proceed with our work [5]. We wanted to integrate taste and health and make it easy for people to cook recipes that are healthy and tasty for them and we are using artificial neural networks for this purpose, because we are dealing with nonlinear equations to predict the taste and the healthiness of the recipe so, all we need is the predicted value. For predicting taste, we need to predict if the person has liked the recipe or disliked it and for predicting healthiness we need to predict if the recipe is acidic or basic and we need to predict if the recipe is a high calorie one or a low calorie one. To get to know if the dish is perfect or not one needs to take feedback from the consumer. The feedback can be a review or a simple like or dislike or a rating. It would be easier for the consumer to just say if he/she liked or disliked the recipe, if he/she wants the recipe to be acidic or not and if he/she wants a low-calorie recipe or not. In this case, the outputs will be taken as 0 s and 1 s since we need to predict like or dislike (depending upon the taste), acidic or basic recipe and a high-calorie recipe or low-calorie recipe which makes it a perfect classification problem. So, we have collected the feedback for various recipes from the consumers and we also recorded taste ratings from people and calories of every dish just in case if we only need the taste rating and calorie content of the recipe. We need not record the P.H values of the dishes since all we want is to predict if a dish is acidic or basic. We recorded the ratings from different people and got different ratings from everyone, since human psychology differs from person to person so does their taste in food so, for predicting taste, we can either use data in which the inputs are ingredients of recipes and the output values are averages of ratings or averages of likes or dislikes which are 0 s and 1 s given by a number of people where an average greater than 0.5 is takes as 1 and below 0.5 is taken as 0 or we can use data in which the inputs are ingredients of recipes with outputs as ratings or with outputs as likes or dislikes which are 0 s and 1 s given by an individual, we have used the data given by an individual for conducting this experiment and the model that we are going to propose in the paper can be used with both the data sets which are data in which outputs are given by an individual and data in which outputs are averages of ratings given by a number of consumers or averages of likes or dislikes which are 0 s and 1 s given by a number of consumers. After collecting the data from

the consumers we have pre-processed the data [6] and then we have designed an algorithm with the idea of using artificial neural networks in food which could make things easier for the chefs and help people to know what to eat and what not to eat.

In this paper, we propose two methodologies which can be very useful to prepare a recipe or suggest a recipe which a fit or a normal person, a person suffering from acidity and a person who desires to lose weight are going to like. This can be a breakthrough in the food industry and can help chefs to prepare a recipe with nearly no flaws, so we have designed a bi-directional [7] LSTMs-based ensemble of neural networks with a very simple gating system and logic gates at the final output end for predicting if a dish is suitable for a specific person or not and for making new recipes we have used a bi-directional LSTMs-based variational autoencoder as a generative model which can make new recipes by varying the flavours or ingredients of the given recipe. After building the bi-directional LSTMs-based ensemble of neural networks, we have used it on 20 new recipes recorded from random people and the model worked exceptionally well on those recipes. This model is used to predict if the person is going to like the dish or not, whether it is good for his health or not and it can also be used to invent new recipes along with bi-directional LSTMs-based variational autoencoder. Two of the few similar previous ensemble neural network models can be seen in [8, 9].

This algorithm can be applied to recipes from any part of the world, the only thing we need is data of recipes and opinions from the consumers. This algorithm is very flexible, it can be used to make different recipes and suggest recipes to people with any kind of health issues by adding as many trained bi-directional LSTMs-based recurrent neural networks as required to the ensemble network. This paper is organized as follows -

- Related works
- Proposed methodologies
- Data and data pre-processing
- Experimental results
- Conclusion

2 Related works

In the present literature, we can find a few works on calorie prediction and the application of deep learning in food. In [10], the authors proposed a methodology, using which we can predict the calorie content of a given food item through the image of the food item by converting the image to a matrix and feeding the matrix to a 'food type classifier model' and a 'size predictor model' after

compressing the matrix. In [11], the author described all the most cited previous works on food recommendation, in which we can find those previous works [12–17] in which the recipe ingredients were used as input features to predict the ratings using basic machine learning models like KNNs, Naive Bayes, Random Forest, SVMs and logistic regression. In [18], the authors have used a deep belief network to predict calories from the food images. In [19], the authors describe how deep learning can be used in estimating the calorie content of the food and the quality of the food. The problem with the previous models is that they are not accurate enough. While predicting taste ratings, there should be a relation among all the ingredients, which can be enabled by using bi-directional LSTMs. While predicting calories, images cannot provide complete information about the recipe as the calorie content of the recipe highly depends upon the cooking time and the combination of ingredients. The precise calorie content of a particular recipe cannot be found by simply summing up the calorie contents of all the ingredients used. Hence, we use an ensemble of bi-directional LSTMs-based framework, which can solve the problem. In [20], the authors use bi-directional LSTMs for word segmentation as it is capable of keeping the contextual information in both directions. In [21], the authors propose a LSTM-based model to infer the production of fractured horizontal wells in a volcanic reservoir, which shows very accurate predictions compared to the traditional models. The LSTM neural network enables to capture the dependencies of the oil rate time sequence data and incorporate the production constraints. Particle Swarm Optimization algorithm was employed to optimize the LSTM model. In [22], the authors propose a hybrid trajectory prediction method based on a sequential model, that fuses two neural networks of a convolutional neural network (CNN) and a long short-term memory network (LSTM). The trajectories of surrounding vehicles are predicted by merging the characteristics of CNN space expansion and LSTM time expansion. Ensemble learning is very effective while dealing with multiple prediction problems, which are mathematically combined to form a resultant output from a given set of input features. In [23], the authors use ResNet, GoogleNet, VGGNet, and InceptionV3 with fine-tuning and ensemble learning for image-based and computer-aided food recognition task, in which they use 'n' number of CNNs for effective prediction. In [24], daily cycling data from eight implantable Li-ion cells were first used as the source dataset to pre-train eight deep convolutional neural network models. The learned parameters of the pre-trained DCNN models are then transferred from the source task to the target task, resulting in eight DCNN with transfer learning (DCNN-TL) models, which are then integrated to build an ensemble model called the DCNN with ensemble learning

and transfer learning (DCNN-ETL). In this model outputs of n individual DCNN-TL models, were received by a fully connected ensemble layer. This layer was used to assign the model weights, and compute an estimated target for the input sample.

3 Proposed methodologies

3.1 Proposed architecture for taste and healthiness prediction

The choice of neural networks is a bi-directional LSTMs-based ensemble of neural networks architecture [25] with logic gates for recipe suggestion. Since the quantity of a particular recipe ingredient needed to make the recipe taste good and healthy for a group of people or a specific person highly depends upon the quantity of previously added ingredient and quantity of the ingredient that's going to be added next, we use bi-directional LSTMs as shown in Fig. 1, which shows the recipe suggestion model. We have used the bi-directional LSTMs among the different types of LSTMs and sequence models as it enables the relationship between the series of ingredients in both the directions and has an optimal number of trainable parameters, which enables the model to perform well with a reasonable training time unlike the other LSTM and sequence models like H-LSTMs [26] and GRUs [27]. Each bi-directional neural network in Fig. 1 classifies a particular aspect of food. The aspect of interest we have taken are a rating of food, the healthiness of food to diabetic, ulcerous, high blood pressure. Diabetic patients need to eat fewer calorie recipes, ulcerous patients need to eat basic recipes (not acidic) and high blood pressure patients need to eat less calorie and basic recipes. Not everyone suffers from the listed health conditions, so we have provided a method for an individual to choose his health condition through a gating system and logic gates, where the neural network's output is passed through the gating system and the logic gates. The bi-directional LSTM-RNN learner 1 will be a classifier model when we want binary output for taste, and it will be a regression model when we want the taste ratings. The bi-directional LSTM-RNN learner 2 will act as a classifier for classifying recipes based on the acidity of the recipe. The bi-directional LSTM-RNN learner 3 will be a classifier model when we want to classify recipes into high and low-calorie consisting recipes, and it will be a regression model when we want to predict the calorie content of the recipe. A threshold value of 0.5 is set at all the 3 binary converters when the bi-directional LSTM-RNN learners are classifiers when the bi-directional LSTM-RNN learner 1 and bi-directional LSTM-RNN learner 3 are regression models, threshold rating value and calorie value needs to be set as per our likes, at binary converter 1 and binary converter 3,

Fig. 1 Bi-directional LSTMs-based ensemble of neural networks

respectively, above which the recipe can be considered as liked recipe and high-calorie recipe. It is futile to use the bi-directional LSTM-RNN learner 3 as a regression model for predicting acidity of the recipe since all we need to do is classify if the dish is acidic or basic, we use it as a classifier, which when predicts a value above 0.5, the recipe is considered to be acidic and vice-versa. Figure 1 shows the proposed architecture of the bi-directional LSTMs-based ensemble of neural networks. The input layers of the neural networks have a shape of (44,1), which we discuss in 3.1, where we discuss the dataset.

Bi-directional LSTM-RNNs are an extension of traditional LSTMs that can improve model performance on classification problems in which sequential data is involved. In this problem, the ingredients, in the dataset are sequential and the quantity of each ingredient that needs to be added highly depends upon the quantity of the previous ingredient and the next ingredient. Using bi-directional LSTMs, we compute 3 main things, those are the backward hidden sequence, the forward hidden sequence, and the output sequence. Equations (1), (2), and (3) give us the equations of the backward hidden sequence, the forward hidden sequence, and the output sequence, respectively.

$$h_t^{\rightarrow(i)} = f(W_{xh}^{\rightarrow} + W_{hh}^{\rightarrow} * h_{t+1}^{\rightarrow} + b_h^{\rightarrow}) \quad (1)$$

$$h_t^{\leftarrow(i)} = f(W_{xh}^{\leftarrow} + W_{hh}^{\leftarrow} * h_{t+1}^{\leftarrow} + b_h^{\leftarrow}) \quad (2)$$

$$y_t = W_{hy}^{\rightarrow} * h_t^{\rightarrow} + W_{hy}^{\leftarrow} * h_t^{\leftarrow} + b_y \quad (3)$$

In Eqs. (1), (2), and (3)-

i is the hidden layer number, f is the hidden layer function, $h_t^{\rightarrow}$ is the forward hidden sequence, $h_t^{\leftarrow}$ is the backward hidden sequence, W_{xh} is the input-hidden weight matrix, W_{hh} is the hidden-hidden weight matrix, W_{hy} is the hidden-output weight matrix, b_h is the bias in the hidden layer, b_y is the bias in the output layer.

The output in a standard neural network (the output layer) is computed as—

$$\hat{y} = \sigma(W^T * X + b) \quad (4)$$

In Eq. (4), $\hat{y}$ is the predicted output, σ is the activation function, W is the weight matrix, X is the input data, and b is the bias of the neural network.

3.1.1 Taste prediction

In Fig. 1 the bi-directional LSTM-RNN learner 1 is a neural network that classifies the recipes based on a person's taste as liked recipes and disliked recipes and it also rates the recipe as per the person's taste. For classifying the

recipes based on taste, this neural network is trained by data consisting of the ingredients of the recipes as the features in the dataset and 0 s and 1 s as output labels in the dataset. The weights of the model are saved after training. 1 in the output label column means the recipe is liked by the person and 0 in the output means the recipe is disliked by the person, an output greater than 0.5 from the bi-directional LSTM-RNN learner 1 means the recipe will be liked by the person. If the output from the bi-directional LSTM-RNN learner 1 is lesser than 0.5, it means the recipe will not be liked by the person. For predicting the taste rating of the recipes, the bi-directional LSTM-RNN is trained with data consisting of the ingredients of the recipes as input features in the dataset and ratings of the recipes as output labels and the weights of the model are saved after training. The output given while predicting the ratings will be a rating of the recipe. 'liked recipes' are those recipes, that will be liked by the person (considering the taste) and 'disliked recipes' are those recipes, that will not be liked by the person.

This neural network while classifying the recipes based on taste has 'binary cross-entropy' [28] as the loss function since we are using the bi-directional LSTM-RNN learner 1 as a classifier, 'RELU' [29, 30] activation function for the input layer and the hidden layers and 'sigmoid' activation function [31] for the output layer. The neural network while predicting the ratings of the recipes can have 'mean squared error' or 'mean absolute error' [32] or the custom loss function that we have designed for this application as a loss function and 'RELU' activation function for the input layer, the hidden layers and the output layer. While compiling this neural network we used 'Adam' as the optimization function for adjusting the weights of the neural network.

The binary cross-entropy loss function is given as-

$$L_{BCE} = -\frac{1}{N} \sum_{i=1}^N (y_i * \log(\hat{y}_i) + (1 - y_i) * \log(1 - \hat{y}_i)) \quad (5)$$

The 'mean squared error' loss function, 'mean absolute error' loss function and the custom loss function we have designed for this application are given as-

$$L_{MSE} = \frac{1}{N} \sum_{i=1}^N (\hat{y}_i - y_i)^2 \quad (6)$$

$$L_{MAE} = \frac{1}{N} * \sum_{i=1}^N |\hat{y}_i - y_i| \quad (7)$$

$$L_{CUSTOM} = \frac{1}{N} \sum_{i=1}^N \frac{|\hat{y}_i - y_i|}{\bar{y}} * 100 \quad (8)$$

where in (1) and (2),

N = size of the sample.

y_i = i th actual value.

$\hat{y}_i$ = i th predicted value.

$\bar{y}$ = average of the ' N ' actual (true) values.

We have used a custom loss function that is similar to 'mean absolute percentage error' [33, 34]. In this the difference between the predicted value and the actual value which are $\hat{y}_i$ and y_i , respectively, is divided by the average of the actual values which is $\bar{y}$ and then we multiply it by 100 to make it a percentage error. We are dividing the difference between $\hat{y}_i$ and y_i by $\bar{y}$ since the ratings given by every person varies, one person might give ratings in the data in such a way that the average of actual values is 7 and the other person might give ratings in the data with an average of actual values as 5 and when predicting the ratings which are averages of the ratings of a group of people the average can be anything so, we need to get an error relative to the value of $\bar{y}$ hence, we have divided the difference between y_i and $\hat{y}_i$ by $\bar{y}$. When the value of $\bar{y}$ is low (around 5 or 6), the loss value computed for particular values of y_i and $\hat{y}_i$, will be higher than the loss value that we get for the same values of y_i and $\hat{y}_i$, when the value of $\bar{y}$ is high (around 8 or 9). If the value of $\bar{y}$ is low then, that means the person is very specific about the taste as he has given good ratings to not many recipes in the dataset, which resulted in a low value of $\bar{y}$. Since the person is specific, the loss value computed even for a small deviation of the predicted value from the true value should be more, when $\bar{y}$ is low and when $\bar{y}$ is high, when the person is not specific in choosing what to eat, the loss computed will be lesser than the loss, that was computed when $\bar{y}$ is low.

The 'RELU' and 'sigmoid' activation functions are given as-

$$f(x) = \max(0, x) \quad (9)$$

$$f(x) = \frac{1}{1 + e^{-x}} \quad (10)$$

where x is the value computed by the previous layer in the neural network.

Equation (11) shows how the weights are updated using Adam [35] optimizer is shown below—

$$w^{t+1} \leftarrow w^t - \eta * \frac{m_t}{\sqrt{v_t + \epsilon}} \quad (11)$$

where w^{t+1} and w^t are updated weights and weights before updating, η is the learning rate, m_t and v_t are updated

biased first-moment estimate and updated bias second raw moment estimate. ϵ is a small scalar, used to prevent division by 0.

3.1.2 Acidic recipe prediction

In Fig. 2 the bi-directional LSTM-RNN learner 2 is an artificial neural network that classifies the recipes based on the acidic content of the recipes since the recipes are cooked the acidity of the recipe is not the same as the acidity of the ingredients required to make the recipe. For classifying the recipes based on the acidic content of the recipe, this neural network is trained by data consisting of the ingredients of the recipe as the features in the dataset and 0 s and 1 s as the output labels in the dataset. 1 in the output means the recipe is acidic and 0 in the output means the recipe is not acidic, an output greater than 0.5 from the bi-directional LSTM-RNN learner 2 means the recipe is acidic. If the output is lesser than 0.5 from the bi-directional LSTM-RNN learner 2, it means the recipe is not acidic.

This neural network while classifying the recipes based on the acidic content has 'binary cross-entropy' as the loss function since we are using the bi-directional LSTM-RNN to classify the recipes, 'RELU' activation function for the input and the hidden layers and 'sigmoid' activation function for the output layer. This neural network also uses 'Adam' as the optimization function for adjusting the weights.

The 'binary cross-entropy' loss function, 'RELU' activation function and 'sigmoid' activation function are given in (1), (5) and (6), respectively.

3.1.3 Calorie prediction

In Fig. 3 the bi-directional LSTM-RNN learner 3 is an artificial neural network that classifies the recipes based on the calories content of the recipes and it also predicts the number of calories in the recipes, since the recipes are cooked the calorie content of the recipe is not the same as the sum of the calories of the ingredients required to make the recipe. For classifying the recipes based on the calorie

Fig. 3 Actual taste ratings of the recipes

content, this neural network is trained by data consisting of the ingredients of the recipe as the features in the dataset and 0 s and 1 s as the output labels in the dataset. 1 in the output means the recipe is high in calorie content and 0 in the output means the recipe has a moderate or low-calorie content, an output greater than 0.5 from the ANN learner 3 means the recipe is high in calorie content. If the output from the bi-directional LSTM-RNN learner 3 is lesser than 0.5, it means the recipe has a moderate or low calorie. For predicting the calorie content of the recipes, the neural network is trained with data consisting of the ingredients of the recipes as input features in the dataset and calories of the recipes as output labels in the dataset, the output is given while predicting the calories of the recipes is the calorie content in the recipes.

This ANN while classifying the recipes based on the calorie content has 'binary cross-entropy' as the loss function since we are using the bi-directional LSTM-RNN learner 3 as a classifier, 'RELU' activation function for the input layer and the hidden layers and 'sigmoid' activation function for the output layer. The neural network while predicting the number of calories of the recipes can have 'mean squared error' or 'mean absolute error' as loss function and 'RELU'

Fig. 2 Bi-directional LSTMs-based VAE

activation function for the input layer, the hidden layers and the output layer. While compiling this neural network we use 'Adam' as the optimization function for adjusting the weights of the neural network.

The 'binary cross-entropy' loss function, 'mean squared error' loss function, 'mean absolute error' loss function, 'RELU' activation function and 'sigmoid' activation function are given in (1), (2), (3), (5) and (6), respectively.

3.1.4 Gating system

The gating system used in the architecture in Fig. 1 is a simple gating system that disconnects the bi-directional LSTM-RNN learner 2 by giving an output 0 from the branch that is connected to the output of the bi-directional LSTM-RNN learner 2 when the person doesn't care about the acidity of the recipe and it disconnects the bi-directional LSTM-RNN learner 3 by giving an output 0 from the branch that is connected to the output of the bi-directional LSTM-RNN learner 3 when the person doesn't care about the calorie content of the recipe.

3.1.5 Logic circuit

We have used a basic logic circuit [36] to get the final output from the ensemble of neural networks which is a suggestion to cook the recipe or not. Both the outputs from the gating system are 0 for an ordinary person with no restrictions on the calorie intake and acidity of the food. There will be 2 possible outcomes from the model which are 0 and 1. We get 0 as the final output when the output of the bi-directional LSTM-RNN learner 1 is 0 and we get 1 as the final output when the output from the bi-directional LSTM-RNN learner 1 is 1. When the person has restrictions on the acidic content of the recipe, the output of the gating system which is connected to the bi-directional LSTM-RNN learner 2 will be 1 and the output of the gating system which is connected to the bi-directional LSTM-RNN learner 3 will be 0. If the person has restrictions on the calorie content of the recipe the output of the gating system connected to the bi-directional LSTM-RNN learner 3 will be 1 and the other output will be 0. If the person has restrictions on both the acidity and the calorie content of the recipe, both the outputs from the gating system will be 1. The person just needs to mention what kind of a dish he is expecting in terms of the acidity of the recipe, the taste of the recipe and the calorie content of the recipe from the ingredients of the recipe. Considering all the cases we have designed a basic logic circuit with an output equation given below.

$$G = a * b' * c' \quad (12)$$

where G is the final output, a is the output from the bi-directional LSTM-RNN learner 1 after passing the Binary input 1 block, b is the output of the bi-directional LSTM-RNN learner 2 after passing the Binary input 2 block and c is the output of the bi-directional LSTM-RNN learner 3 after passing the Binary input 3 block. The circuit is designed in such a way that the final output will be 1 only if the values of $a = 1$, $b = 0$ and $c = 0$. The final output is the suggestion which can be either 0 or 1 when the final output is 0, the recipe is not suggestible to that person but when the final output is 1, the recipe is suggestible to that person.

3.2 Proposed bi-directional LSTMs-based variational autoencoders for generating new recipes

Variational Autoencoders (VAEs) have emerged as one of the most popular approaches to unsupervised learning of complicated distributions. VAEs are appealing, because they are built on top of standard function approximators (neural networks). VAEs are used for reconstructing inputs, encode inputs into latent variables, and generate data from the latent variables [37]38. GANs are also used as generative models, which are mostly used for the reconstruction of images. In this case, for generating recipes, we are going to use bi-directional LSTMs-based VAEs. We use bi-directional LSTMs as a generative model here as, every ingredient in a recipe, depends upon the previous and the next ingredient of the recipe. Bi-directional LSTMs enable the relations between ingredients, which is very critical as the taste of a recipe directly depends upon the nonlinear combination of the ingredients used to cook the dish and due to this, the model would be able to predict more appropriate dishes for the person. Figure 2 shows the architecture of the bi-directional LSTMs-based VAE that we have used, to generate new recipes, from the existing ones.

The bi-directional LSTMs-based VAE shown in Fig. 1 is fed with data of those recipes that are fit for the person, which are the suggested recipes, suggested by the bi-directional LSTMs-based ensemble of neural networks shown in Fig. 1. If the model is fed with data of those recipes that are not healthy for the person and not liked by the person, considering the taste, then the generative model would generate those recipes that are healthy for the person and not tasty for the person. The neural networks in Figs. 1 and 2 can be trained with the data collected from a group of people (with the outputs in the dataset as averages of the outputs given by the group of people) or a specific person, depending upon the usage of the model. The pre-processing of the data, before feeding into the models, shown in Figs. 1 and 2, is discussed in the next section.

Equation (13) is the equation used for the implementation of the sampling operation of 'z' (equation used for sampling)-

$$z = \mu(x) + \sigma(x)^{1/2} * \epsilon \quad (13)$$

In the above equation, z is the latent variable, $\mu(x)$ and $\sigma(x)$ are the Gaussian parameters [39], and $\epsilon \sim N(0,1)$.

The re-constructual loss of VAE is given in Eq. (14), the KL loss of VAE is shown in Eq. (15), and the complete loss of VAE used is given in Eq. (16)-

$$L_{\text{recon}} = \sum_k (y_i * \log(\hat{y}) + (1 - y_i) * \log(1 - \hat{y})) \quad (14)$$

$$L_{\text{KL}} = \frac{1}{2} \sum_k (\exp(\sigma(x)) + \mu^2(x) - 1 - \sigma(x)) \quad (15)$$

$$L_{\text{final}} = L_{\text{recon}} + L_{\text{KL}} \quad (16)$$

The optimization function that was used to update weights used was 'Adam', given in Eq. (11). We have used 'RELU' activation function in all the layers but for the final layer, where we used 'sigmoid' activation function. We get output values ranging from 0 to 1, from the bidirectional LSTMs-based VAE. We get the new recipe after multiplying the output values with twice the amount of respective existing recipe ingredients, which were given as inputs to the VAE.

4 Data and data pre-processing

4.1 Dataset

Dataset is a collection of data where data numerically defines the Observations collected. We have collected data for predicting taste (like or dislike), acidity, and calorie content from various Cooking websites [40–43] and their ratings at various restaurants from various people. We have also collected data to classify recipe's edibleness for diabetic, peptic ulcer, high blood pressure patients. The recipes map to 38 ingredients. The ingredients are Indian cuisine oriented (onions, ginger garlic, green chillies, curd, cashews, milk solids, flavour leaf, cloves, cinnamon cardamom, mace, caraway, turmeric, red chilli, coriander, garam masala, cream, water, pudina, ghee, salt, lime, cornflour, oil, onion, soya sauce, tomatoes, tomato sauce, vinegar, cumin, coconut, chat masala, sugar, lemon, butter, methi, tamarind, pepper) these ingredients are chosen from recipe websites [40–43]. The features of the dataset are shown in Table 1. The dataset with input features columns and output columns, maps

Table 1 Names of the ingredients

Ingredient Number	Ingredient	Measure/ Represented by
1	Main ingredient name	Name
2	Main ingredient	Grams
3	Onions	Cups
4	Ginger Garlic	Spoons
5	Green Chilli	Cups
6	Curd	Cups
7	Cashews	Cups
8	Milk solids	Grams
9	Flavour leaves	Count
10	Cloves	Count
11	Cinnamon	Pinches
12	Cardamom	Spoons
13	Mace	Spoons
14	caraway	Spoons
15	turmeric	Spoons
16	Red chilli powder	Spoons
17	Coriander	Spoons
18	garam masala	Spoons
19	Cream	Spoons
20	Water	Cups
21	Pudina	Cups
22	Ghee	Spoons
23	Salt	Spoons
24	Time	Minutes
25	cornflour	Cups
26	Oil	Spoons
27	spring onion	Spoons
28	soya sauce	Spoons
29	tomatoes	Cups
30	vinegar	Spoons
31	Cumin	Spoons
32	coconut	Cups
33	chat masala	Spoons
34	Sugar	Spoons
35	Lemon	Spoons
36	Butter	Spoons
37	Methi	Cups
38	tamarind	Spoons
39	pepper	Spoons

to a shape of (number of samples, 42), in which the input features columns are 39 and the output columns are 3 (taste, acidic recipe, high-calorie recipe). The input layers of the neural networks used have 44 input neurons as the dataset with input features only, has a shape of (number of samples, 39) and it becomes a dataset with

shape (number of samples, 44) after data pre-processing. Then the 2D data is reshaped into 3D data before feeding it into the model. In the model shown in Fig. 2, the dataset remains with the shape (number of samples, 39) and it is converted from 2D to 3D. We discuss data pre-processing in 4.2.

4.2 Data pre-processing

Data pre-processing is a significant process to generalize the performance of the neural network. It eliminates the noise and deals with the missing values in the dataset [44]. For pre-processing the data to feed it into the model shown in Fig. 1, we have used label encoder and one-hot encoder from the scikit learn [6] to pre-process the data and then eliminated the first column to remove the dummy variable trap. Since we had 6 different main ingredients in the dataset, on using the label encoder on the main ingredients column, we get numbers 1 to 6 in the main ingredient column, which represent those 6 main ingredients. Now we use one-hot encoder on the data which enables the representation of categorical data to be more expressive, this produces 6 different columns for representing 6 different main ingredients, which were transformed into numbers (1 to 6) by the label encoder. After those two steps, we remove the first column to avoid the dummy variable trap [45] and then we have scaled the data using standard scalar from the scikit learn library to scale the data before feeding it into the neural networks, after pre-processing each input mapped to a shape of (number of samples, 44). Then we reshape the 2D data into 3D data with shape (number of samples, 44, 1) and we feed it into the model shown in Fig. 1. For pre-processing the data to feed it into the model shown in Fig. 2, we have used the label encoder on the main ingredients column and this gives us numbers 1 to 6, in the main ingredients column and the dataset remains with the shape (number of samples, 39). After using the label encoder, we reshape the 2D data into 3D data with shape (number of samples, 39, 1) and feed it into the model. 'Main ingredient 1', 'main ingredient 2', 'main ingredient 3', 'main ingredient 4',

'main ingredient 5' and 'main ingredient 6' were label encoded into '1', '2', '3', '4', '5' and '6', respectively.

5 Results and discussions

5.1 Classification results

We have trained and tested the model with 3000 recipes (number of samples), and we have collected 20 unknown recipes and tested the performance of the model on those unknown recipes. To evaluate the performance of the bi-directional LSTM-RNN learners while they are classifiers, we have plotted the confusion matrix and calculated the performance parameters which are precision, recall, F_1 -score, test negative rate, false-positive rate and accuracy. Table 2 shows the confusion matrix of actual liked or disliked recipes and predicted liked and disliked recipes. Table 3 shows the confusion matrix of the actual acidic or non-acidic recipes and predicted acidic or non-acidic recipes. Table 4 shows the confusion matrix of actual high calorie or low-calorie recipe and predicted high calorie and low-calorie recipe. Table 5 shows us the performance parameters of the bi-directional LSTM-RNN learners. Since there are no other existing models which predict the taste, acidic nature of a recipe, and calorie content of a recipe, we have plotted the results of only our model.

The performance comparison of the model with KNNs [46], SVMs [47], standard neural networks [48], LSTMs [49] (which were the state-of-art models in [12–17]) in place of bi-directional LSTMs-based RNN is shown in Tables 6, 7, 8, 9, respectively.

Table 2 Confusion matrix while classifying the recipes into liked and disliked recipes (binary classification based on taste)

	Predicted Positive (acidic)	Predicted Negative (non-acidic)
Actual positive (acidic)	187 (TP)	56 (FN)
Actual Negative (non-acidic)	32 (FP)	525 (TN)

Table 3 Confusion matrix while classifying the recipes into acidic and non-acidic(basic)

	Predicted Positive (like)	Predicted Negative (dislike)
Actual Positive (like)	626 (TP)	72 (FN)
Actual Negative (dislike)	28 (FP)	74 (TN)

Table 4 Confusion matrix while classifying the recipes into high-calorie recipes and moderate or low-calorie recipes

	Predicted Positive (high calorie)	Predicted Negative (moderate and low calorie)
Actual positive (high calorie)	371 (TP)	38 (FN)
Actual Negative (moderate and low calorie)	63 (FP)	328 (TN)

Table 5 Performance parameters of the bi-directional LSTM-RNN learners

S. No	Performance Parameter	Bi-directional LSTM-RNN learner 1	Bi-directional LSTM-RNN learner 2	Bi-directional LSTM-RNN learner 3
1	Recall	0.896	0.899	0.896
2	Precision	0.957	0.736	0.864
3	F ₁ -score	0.925	0.809	0.879
4	True Negative rate	0.725	0.942	0.838
5	False Positive rate	0.274	0.057	0.161
6	Accuracy	0.875	0.890	0.873

Table 6 Performance parameters of the KNN learners

S. No	Performance Parameter	KNN learner 1	KNN learner 2	KNN learner 3
1	Recall	0.855	0.802	0.829
2	Precision	0.914	0.684	0.792
3	F ₁ -score	0.884	0.739	0.810
4	True Negative rate	0.451	0.838	0.772
5	False Positive rate	0.549	0.162	0.228
6	Accuracy	0.804	0.828	0.801

Table 7 Performance parameters of the SVM learners

S. No	Performance Parameter	SVM learner 1	SVM learner 2	SVM learner 3
1	Recall	0.867	0.848	0.792
2	Precision	0.919	0.684	0.862
3	F ₁ -score	0.892	0.757	0.825
4	True Negative rate	0.480	0.829	0.867
5	False Positive rate	0.520	0.171	0.133
6	Accuracy	0.818	0.835	0.829

Table 8 Performance parameters of the standard neural network learners

S. No	Performance Parameter	Standard Neural Network learner 1	Standard Neural Network learner 2	Standard Neural Network learner 3
1	Recall	0.864	0.856	0.773
2	Precision	0.942	0.693	0.895
3	F ₁ -score	0.901	0.766	0.829
4	True Negative rate	0.637	0.835	0.905
5	False Positive rate	0.363	0.165	0.095
6	Accuracy	0.835	0.841	0.838

Table 9 Performance parameters of the LSTM learners

S. No	Performance Parameter	LSTM learner 1	LSTM learner 2	LSTM learner 3
1	Recall	0.883	0.852	0.809
2	Precision	0.945	0.772	0.887
3	F ₁ -score	0.913	0.810	0.847
4	True Negative rate	0.647	0.890	0.893
5	False Positive rate	0.353	0.110	0.107
6	Accuracy	0.853	0.879	0.850

Table 10 Actual values and predicted by the bi-directional LSTM-RNN learners during classification

Recipe num	Actual likes and dislikes	Actual acidic recipe	Actual high-calorie recipe	Likes and dislikes prediction (considered as 1 if value is > 0.5 else, considered as 0)	Acidic recipe prediction (considered as 1 if value is > 0.5 else, considered as 0)	High-calorie prediction (considered as 1 if value is > 0.5 else considered as 0)
01	1	1	0	0.979	0.925	0.340
02	1	1	1	0.730	0.533	0.570
03	1	1	0	0.681	0.544	0.343
04	1	0	0	0.411	0.405	0.292
05	0	1	0	0.286	0.606	0.229
06	0	1	0	0.361	0.705	0.040
07	0	0	0	0.383	0.419	0.254
08	1	1	0	0.735	0.753	0.349
09	0	0	0	0.361	0.353	0.395
10	0	0	0	0.472	0.405	0.179
11	1	1	1	0.687	0.824	0.592
12	1	1	0	0.667	0.765	0.468
13	0	1	0	0.346	0.881	0.286
14	0	0	0	0.467	0.499	0.212
15	0	0	0	0.223	0.305	0.130
16	1	0	0	0.532	0.287	0.205
17	1	0	0	0.598	0.374	0.180
18	1	1	0	0.985	0.883	0.317
19	0	1	0	0.214	0.886	0.185
20	1	1	0	0.592	0.688	0.293

We have tested the classification model on 20 unknown recipes. Table 10 shows the actual likes and dislikes, actual acidic recipes and actual high-calorie recipes along with the predicted likes and dislikes, predicted acidic recipes and predicted high-calorie recipes. The predicted values are outputs of the bi-directional LSTM-RNN learners which are considered as 0 when the output from the bi-directional LSTM-RNN learner is lesser than 0.5 and it is considered to be 1 if the output from the bi-directional LSTM-RNN learner is greater than 0.5.

After the bi-directional LSTM-RNN learner 1, 2 and 3 predict the values, they are automatically converted to 0 s and 1 s at the binary input converter block before going to the logic circuit in Fig. 1. The final outputs of the model or suggestions for those 20 unknown recipes are shown in Table. 11.

5.2 Taste rating prediction and calorie content prediction

5.2.1 Taste rating prediction

The outputs that we got in classification were binary outputs, for taste ratings and calorie content we need to train the bi-directional LSTM-RNN learners 1 and 3 to predict the taste ratings and calorie content. We can use a

regression model instead of a classification model to get the exact ratings and the calorie content, which will be automatically converted into 0 s and 1 s after the output passes the binary input 1 block and binary input 3 blocks. Figure 3 show the actual taste ratings. Figure 4 shows the predicted taste ratings with 'MSE' or 'mean squared error'. Figure 5 shows the predicted taste ratings with 'MAE' or mean absolute error. Figure 6 shows the predicted taste ratings with a custom loss function.

5.2.2 Calorie prediction

We have tested the bi-directional LSTM-RNN learner 3 for predicting the calorie values with the same 20 unknown recipes which were used in like recipe, acidic recipe and high-calorie recipe classification. Figure 7 shows the actual calorie values of those 20 recipes. Figure 8 shows the predicted calorie content of the 20 recipes which looks accurate and close to Fig. 7 which shows the actual calorie content of those 20 recipes.

5.3 Generation of recipes using bi-directional LSTMs-based VAEs

We have trained the variational autoencoder with those recipes that are liked by a specific person to generate

Table 11 The suggestions after binary conversion and passing through the logic circuit

Recipe num	Like	Acidic nature	Calorie content	Suggestion (person with no restrictions)	Suggestion (person with restrictions on acidity)	Suggestion (person with restrictions on calories)	Suggestion (person with both acidity and calories restriction)
01	1	1	0	1	0	1	0
02	1	1	1	1	0	0	0
03	1	1	0	1	0	1	0
04	0	0	0	0	0	0	0
05	0	1	0	0	0	0	0
06	0	1	0	0	0	0	0
07	0	0	0	0	0	0	0
08	1	1	0	1	0	1	0
09	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0
11	1	1	1	1	0	0	0
12	1	1	0	1	0	1	0
13	0	1	0	0	0	0	0
14	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0
16	1	0	0	1	1	1	1
17	1	0	0	0	1	1	1
18	1	1	0	1	0	1	0
19	0	1	0	0	0	0	0
20	1	1	0	1	0	1	0

Fig. 4 Predicted taste ratings with 'MSE' or 'mean squared error' **Fig. 5** Predicted taste ratings with 'MAE' or 'mean absolute error'

recipes for that person as per that person's taste or a group of people to generate recipes for the group of people as per their taste. We have generated 5 new paneer recipes from 5 of the existing paneer recipes. Figure 9 shows the ingredients of existing recipes and Fig. 10 shows the ingredients that were generated to make a new recipe. New recipe 1, new recipe 2, new recipe 3, new recipe 4 and new recipe 5 are the new recipes that were generated from the existing recipe 1, existing recipe 2, existing

recipe 3, existing recipe 4 and existing recipe 5, respectively. The main ingredient, which is paneer in all these recipes weighed 240 to 250 g and while plotting the below graphs we scaled the weight of the main ingredient, which is 250 g in his case, to 1 for better graphs. We also plotted time in hours rather than plotting it in minutes for better graphs.

Fig. 6 Predicted taste ratings with custom loss function **Fig. 7** Actual calories of the 20 recipes **Fig. 8** Predicted calorie content of the 20 recipes **Fig. 9** Ingredients of existing recipes **Fig. 10** Ingredients of generated new recipes **Fig. 11** Results of ablation experimentation

5.4 Ablation experiments

We evaluated the input features in the dataset by conducting an ablation experiment. We have tested 40 different configurations of the dataset. The first configuration of the dataset consists of all the input features. The next 39 configurations are short of one input feature each. Figure 11 shows the results of the ablation experiment, that was conducted. We have used the Binary Cross-entropy loss function for conducting this test. In Table 11 we have used numbers to represent ingredients (input features), which are the serial numbers of the ingredients shown in Table. 1, to represent the respective input feature. The 39 configurations which are short of an input feature each are depicted as 'All features—X', where 'X' is the serial number of the input feature, which the configuration should be short of. 'X' is given as, 'X' = {1, 2, 3..., 39} and 'All features' is the dataset configuration with all the input features. The results

in Fig. 11 shows the importance of each feature in the dataset.

In the figure above, you can see the losses with different configurations. The configuration with the highest loss value has the most important input feature missing and the configuration with the least value of loss has the least important input feature missing in it. On arranging the loss values in descending order, along with the missing respective missing feature, we get the sequence of input features, starting with the most significant input feature to the least significant input feature. We have not used any feature filtering technique while choosing the input features as all the ingredients influence the taste and other properties of a dish.

5.5 Discussions

We have plotted the accuracy of the bi-directional LSTM-RNN learners with the training and the validation data during binary classification. Figure 12 shows the accuracy

Fig. 12 Accuracy of bi-directional LSTM-RNN learner 1 while classifying the recipes based on taste (left), Accuracy of the bi-directional LSTM-RNN learner 2 while classifying recipes based on acidity (mid-

dle), Accuracy of the bi-directional LSTM-RNN learner 3 while classifying recipes based on calorie content (right)

Fig. 13 Loss of bi-directional LSTM-RNN learner 1 while predicting taste ratings with 'MSE' (left), 'MAE' (middle) and custom loss function(right)

of the bi-directional LSTM-RNN learners while classifying the recipes based on taste or likes and dislikes, acidity and calorie content.

We have plotted the loss graphs of the bi-directional LSTM-RNN learner 1 while predicting the taste ratings with 'MAE' or 'mean squared error', 'MAE' or 'mean absolute error' and the custom loss function that we have designed for this problem with the training and the validation data. Figure 13 shows the accuracy of the bi-directional LSTM-RNN learner 1 with different loss functions.

We have also plotted the loss of the bi-directional LSTM-RNN learner 3 while predicting the calorie content of a recipe. Figure 14 shows us the loss of the bi-directional LSTM-RNN learner 3 with the training and the validation data.

We have plotted the training loss and the validation loss that was computed while training the VAE for generating recipes. We have plotted the training and the validation loss of different VAE models; Fig. 15 shows the

Fig. 14 Loss of the bi-directional LSTM-RNN learner 3 while predicting calorie content

Fig. 15 Standard neural VAE performance (top left), LSTM-VAE performance (top right), BI-LSTM VAE performance (bottom)

performance of different VAE models while being trained for generating recipes.

6 Conclusion

In this paper, we have proposed an ensemble neural network architecture which can classify the recipes based on its taste, acidity and calories and we propose a variational autoencoder which is used as a generative model to generate new recipes from the existing standard recipes. The results of the models that we have proposed provide the state-of-art results for classifying the recipes based on its taste, acidity and calories with good accuracies. To evaluate our classification results, different performance evaluation parameters such as precision, recall, F-score, false-positive rate, true negative rate and the performance of the model with 20 unknown recipes have been included. To see the working of the model while the bi-directional LSTM-RNN learner 1 and bi-directional LSTM-RNN learner 3 are acting as regression models to predict taste ratings and calorie content, respectively, we have shown the results that we have got during taste rating prediction and calorie content prediction for those 20 unknown recipes. To see the working of the variational autoencoder model we have generated 2 new recipes from 2 existing recipes. Considering the uniqueness of this ensemble artificial neural network which can suggest recipes based on taste, acidity and calorie content, more researches will be done to make this model better in the future.

Acknowledgements This research was funded by Chaitanya Bharathi Institute of Technology, Telangana, India.

Declarations

Conflict of interest The author(s) declare that they have no competing interests

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing,

adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Prasad NN, Siddalingaswamy M, Parameswariah PM, Radhakrishna K, Rao RV, Viswanathan KR, Santhanam K (2000) Proximate and mineral composition of some processed traditional and popular Indian dishes. *Food chem* 68(1):87–94
2. Bhatia SJ, NageshwarReddy D, Ghoshal UC, Jayanthi V, Abraham P, Choudhuri G, Broor SL et al (2011) Epidemiology and symptom profile of gastroesophageal reflux in the Indian population: report of the Indian Society of Gastroenterology Task Force. *Indian J Gastroenterol* 30(3):118
3. Salvador A, Drozdal M, Giro-i-Nieto X, Romero A (2019) Inverse cooking: Recipe generation from food images. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 10453–10462
4. Salvador A, Hynes N, Aytar Y, Marin J, Ofli F, Weber I, Torralba A (2017) Learning cross-modal embeddings for cooking recipes and food images. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 3020–3028
5. Cruz, ELA (2015) Artificial neural network implementation for prediction of beer taste quality
6. Géron, A (2019) *Hands-on machine learning with scikit-learn, keras, and tensorflow: concepts, tools, and techniques to build intelligent systems*. O'Reilly Media, 2019
7. Gandhi N, Petkar O, Armstrong LJ (2016) Rice crop yield prediction using artificial neural networks. In: *2016 IEEE Technological Innovations in ICT for Agriculture and Rural Development (TIAR)*, pp 105–110. IEEE
8. Jovanović RŽ, Sretenović AA, Živković BD (2015) Ensemble of various neural networks for prediction of heating energy consumption. *Energy Build* 94(2015):189–199
9. Lee H, Hong S, Kim E (2009) Neural network ensemble with probabilistic fusion and its application to gait recognition. *Neurocomputing* 72(7–9):1557–1564

10. Chokr M, Elbassuoni S (2017) Calories prediction from food images. In: Proceedings of the thirty-first AAAI conference on Artificial Intelligence (AAAI'17). AAAI Press, pp 4664–4669
11. Trattner C, Elswailer D (2017) Food recommender systems: important contributions, challenges and future research directions." arXiv preprint arXiv:1711.02760
12. Elswailer D, Hors-Fraile S, Ludwig B, Said A, Schäfer H, Trattner C, Orkamaan H, Valdez AC (2017) Second workshop on health recommender systems: (healthrecsys 2017). In: Proceedings of the eleventh ACM conference on recommender systems, recsys 2017, como, italy, august 27–31, 2017 (pp 374–375)
13. Elswailer D, Trattner C, Harvey M (2017) Exploiting food choice biases for healthier recipe recommendation. In: Proceedings of the 40th international acm sigir conference on research and development in information retrieval (pp 575–584).
14. Elswailer D, Harvey M, Ludwig B Said A (2015) Bringing the "healthy" into food recommenders. In: Dmrs (pp 33–36)
15. Harvey M, Ludwig B, Elswailer D (2013) You are what you eat: Learning user tastes for rating prediction. In: Proceedings of the 20th international symposium on string processing and information retrieval—volume 8214 (pp 153–164).
16. Teng C-Y, Lin Y-R, Adamic LA (2012) Recipe recommendation using ingredient networks. In: Proceedings of the 4th annual ACM web science conference (pp 298–307).
17. Freyne J, Berkovsky S (2010) Intelligent food planning: Personalized recipe recommendation. In: Proceedings of the 15th international conference on intelligent user interfaces (pp 321–324)
18. Pouladzadeh P, Kuhad P, Peddi SVB, Yassine A, Shirmohammadi S (2016) Food calorie measurement using deep learning neural network. 1–6. <https://doi.org/10.1109/I2MTC.2016.7520547>.
19. Zhou L, Zhang C, Liu F, Qiu Z, He Y (2019) Application of deep learning in food: a review. *Compr Rev Food Sci Food Saf* 18:1793–1811
20. Yao Y, Zheng H (2016) Bi-directional LSTM recurrent neural network for Chinese word segmentation. In: International conference on neural information processing, pp 345–353. Springer, Cham
21. Song X, Liu Y, Xue L, Wang J, Zhang J, Wang J, Jiang L, Cheng Z (2020) Time-series well performance prediction based on Long Short-Term Memory (LSTM) neural network model. *J Petrol Sci Eng* 186:106682
22. Xie G, Shangguan A, Fei R, Ji W, Ma W, Hei X (2020) Motion trajectory prediction based on a CNN-LSTM sequential model. *Sci China Inf Sci* 63(11):1–21
23. Tasci E (2020) Voting combinations-based ensemble of fine-tuned convolutional neural networks for food image recognition. *Multimed Tools Appl* 79(41):30397–30418
24. Shen S, Sadoughi M, Li M, Wang Z, Chao Hu (2020) Deep convolutional neural networks with ensemble learning and transfer learning for capacity estimation of lithium-ion batteries. *Appl Energy* 260:114296
25. Bahad P, Saxena P, Kamal R (2019) Fake news detection using bi-directional LSTM recurrent neural network. *Proc Comput Sci* 165:74–82
26. Dai X, Yin H, Jha NK (2019) Grow and prune compact, fast, and accurate LSTMs. *IEEE Trans Comput* 69(3):441–452
27. Chung J, Gulcehre C, Cho KH, Bengio Y (2014) Empirical evaluation of gated recurrent neural networks on sequence modeling." arXiv preprint arXiv:1412.3555
28. Ramos D, Franco-Pedroso J, Lozano-Diez A, Gonzalez-Rodriguez J (2018) Deconstructing cross-entropy for probabilistic binary classifiers. *Entropy* 20(3):208
29. Schmidt-Hieber J (2017) Nonparametric regression using deep neural networks with ReLU activation function." arXiv preprint arXiv:1708.06633
30. Eckle K, Schmidt-Hieber J (2019) A comparison of deep networks with ReLU activation function and linear spline-type methods. *Neural Netw* 110:232–242
31. Zhang H, Tsui-Wei W, Pin-Yu C, Cho-Jui H, Luca D (2018) Efficient neural network robustness certification with general activation functions. In: Advances in neural information processing systems, pp 4939–4948
32. Beale HD, Demuth HB, Hagan MT (1996) Neural network design. Pws, Boston
33. De Myttenaere A, Golden B, Le Grand B, Rossi F (2016) Mean absolute percentage error for regression models. *Neurocomputing* 192(2016):38–48
34. Khair, U, Fahmi H, Al Hakim S, Rahim R (2017) Forecasting error calculation with mean absolute deviation and mean absolute percentage error. In: *Journal of Physics: Conference Series*, vol 930, no. 1, p 012002. IOP Publishing
35. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980
36. Shiva SG (ed) (2018) Introduction to logic design. CRC Press, Cambridge
37. Wan Z, Zhang Y, He H (2017) Variational autoencoder based synthetic data generation for imbalanced learning. In: 2017 IEEE symposium series on computational intelligence (SSCI), pp 1–7. IEEE
38. Semeniuta S, Severyn A, Barth E (2017) A hybrid convolutional variational autoencoder for text generation. arXiv preprint arXiv:1702.02390
39. Ma C, Gong W, Hernández-Lobato JM, Koenigstein N, Nowozin S, Zhang C (2018) Partial VAE for hybrid recommender system. In: NIPS Workshop on Bayesian Deep Learning
40. <https://www.allrecipes.com/>
41. <https://www.indianhealthyrecipes.com/recipes/veg-curry/>
42. <https://in.pinterest.com/sw-shell.html>
43. <https://www.cookwithmanali.com/category/recipes/indian-main-course-vegetarian-recipes/>
44. Kotsiantis SB, Kanellopoulos D, Pintelas PE (2006) Data preprocessing for supervised learning. *Int J Comput Sci* 1(2):111–117
45. Pal M, Bharati P (2019) The regression models with dummy explanatory variables. In: Applications of regression techniques, pp 135–153. Springer, Singapore
46. Guo G, Wang H, Bell D, Bi Y, Greer K (2003) KNN model-based approach in classification. In: OTM Confederated International Conferences" On the Move to Meaningful Internet Systems, pp 986–996. Springer, Berlin
47. Shon T, Moon J (2007) A hybrid machine learning approach to network anomaly detection. *Inf Sci* 177(18):3799–3821
48. Sarle, WS (1994) Neural networks and statistical models
49. Srivastava N, Mansimov E, Salakhudinov R (2015) Unsupervised learning of video representations using lstms. In: International conference on machine learning, pp 843–852

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