Text classification to predict skin concerns over skincare using bidirectional mechanism in long short-term memory

Devi Fitrianah[1], Andre Hangga Wangsa[2]
[1]Department of Computer Science, School of Computer Science, Bina Nusantara University, Jakarta, Indonesia
[2]Department of Informatics, Faculty of Computer Science, Universitas Mercu Buana, Jakarta, Indonesia

Article Info

Article history:

Received Mar 10, 2022
Revised Jul 29, 2022
Accepted Sep 12, 2022

Keywords:

Dermatology
Multi-class text classification
deep learning
Natural language processing
Skincare

ABSTRACT

There are numerous types of skincare, each with its own set of benefits based on key ingredients. This may be difficult for beginners who are purchasing skincare for the first time due to a lack of knowledge about skincare and their own skin concerns. Hence, based on this problem, it is possible to find out the right skin concern that can be handled in each skincare product automatically by multi-class text classification. The purpose of this research is to build a deep learning model capable of predicting skin concerns that each skincare product can treat. By comparing the performance and results of predicting the correct skin condition for each skincare product description using both long short-term memory (LSTM) and bidirectional long short-term memory (Bi-LSTM), The best results are given by Bi-LSTM, which has an accuracy score of 98.04% and a loss score of 19.19%. Meanwhile, LSTM results have an accuracy score of 94.12% and a loss score of 19.91%.

Corresponding Author:

Andre Hangga Wangsa
Department of Informatics, Faculty of Computer Science, Universitas Mercu Buana
Jalan Raya Meruya Selatan no. 1, Kembangan Jakarta Barat-16550, Indonesia
Email: andrehanggaw@gmail.com

1. INTRODUCTION

Skincare products are the most mainstream cosmetics that maintain skin integrity, appearance, and condition. The high market demand has made skincare products one of the most popular ways to deal with skin concerns [1]. What's more, skincare trends began to rise drastically in 2020, when the COVID-19 pandemic began [2].

Skincare has various types and benefits according to the active ingredients contained in it [3]. Active ingredients here play an important role in the performance of every skincare product because these ingredients are chemicals that actively work on a specific target skin concern [4]. For example, salicylic acid can reduce sebum secretion so that it can control oily skin and acne, but on the other hand, it can also cause inflammation in sensitive and dry skin [5]. This is what makes skincare products difficult to use and unsuitable for beginners; the user must thoroughly understand what is contained in them in order to meet their skin care concerns and expectations [6].

Most beauty stores already manually sort all of their skincare products based on brands, skin types, and skin concerns. But it will take a long time and require someone who knows about skincare products. Instead, by collecting all information that relates to skincare products, such as the function of the product in dealing with certain skin concerns, we might be able to build a model that can automatically classify and predict the benefits of those skincare products quickly.

The information given to classify and predict is in the form of text data, which is a description of skincare products, so it is called a text classification. Text classification is one of the tasks in natural language processing (NLP), which aims to assign labels or targets to textual features or classes such as sentences, queries, paragraphs, and documents [7]. There are two problems in text classification: binary and multi-class classification. Binary classification is made up of only two labels, one of which is assigned a value in an arbitrary feature space X [8]. Whereas multi-class classification has more than two labels [9].

There are various kinds of research on multiclass classification problems, despite the different uses of domains or topics, data types, and algorithms. Although there is currently no research related to skin care products, there are several studies that discuss the dermatology domain. The goal of Indriyani and Sudarma's [3] study was to categorize facial skin types, which were divided into four categories: normal, dry, oily, and combination skin. They used an image-type dataset of sixty facial images captured manually with a digital camera. Although this makes it into computer vision instead of NLP, at least with a case that aims to classify multiclass facial skin types and also by using a supervised learning algorithm, support vector machine (SVM), The result is an average accuracy score of 91.66% and an average running time of 31.571 seconds, which is higher than in previous studies [10]. The following study made extensive use of deep neural network algorithms such as convolutional neural network (CNN), recurrent neural network (RNN), and long short-term memory (LSTM) [11]. Although there are a few cases of binary classification because some datasets only have two classes, the majority of the datasets have between five and ten classes. The research combines several of those algorithms into a hybrid framework. Not only that, some algorithms are also modified into a bidirectional mechanism. The proposed model achieved excellent performance on all tasks. A bidirectional recurrent convolutional neural network attenuation-based (BRCAN) gave accuracy scores on the four multi-class classification tasks of 73.46%, 75.05%, 77.75%, and 97.86%; those results are higher than all comparison algorithms.

In relation to the aforementioned studies, we proposed a comparison of unidirectional/regular LSTM and bidirectional long short-term memory (Bi-LSTM) in our own dataset collected from several skincare online stores to classify skin concerns of each skincare product. The main purpose of this research is to find out the difference between the performance results of the two proposed algorithms. In other research, bidirectional mechanisms, which have layers that work forward and backward in sequence, are able to outperform unidirectional LSTM [12].

2. METHOD

This section of the paper presents the research methodology. When doing research, researchers must obtain data that will be studied for later processing. After obtaining the data, the data is still in the form of raw data, which then the researcher must prepare the data to become a data set that can be processed. After the data has gone through the processing stage, the last stage the researcher must do is to evaluate the research model or instrument to understand their performance, as well as its strengths and weaknesses. More details, there are several stages can be seen in Figure 1.

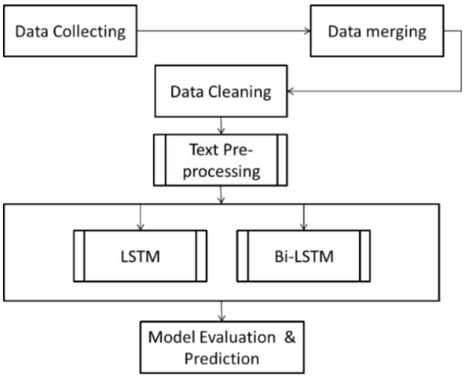


Figure 1. The proposed research methodology

2.1. Data collection

In this research, data collection was implemented by using the Web Scraping technique. Web Scraping is used to convert unstructured data into structured data that can be stored and analyzed in a central local database or spreadsheet [13]. The data is collected on a beauty online store website which is lookfantastic.com, dermstore.com, allbeauty.com, sokoglam.com, and spacenk.com which market products such as skincare, makeup, and beauty tools.

2.2. Data merging

The data collected are divided into three categories according to the seven skin concerns handled by each skincare product, due to having similar symptom treatment. The three categories are drynes, redness; anti-aging, wrinkles; acne, big pores, blemish. Data has 6 attributes which is skincare name, skincare price, how to use, skin concerns, product description, ingredients, and active ingredients. The data that has been collected is merged into one dataset with a total of 5183 rows.

2.3. Data cleaning

Due to data that has the same value (duplicate data) and data that has no value (null data), then data cleaning is carried out by removing duplicate and null data evenly. Data cleaning greatly improves the accuracy of machine learning models, which however requires broad domain knowledge to identify examples that will influence the model [14]. After that, the total dataset is reduced to 5152 rows. However, in this study, we will focus on the attributes of product descriptions that will become features and skin problems that will become labels. So, we will delete the other columns that are not necessary to make the process easier going forward.

2.4. Text preprocessing

Before fed the dataset to our models, it's necessary to perform a data pre-processing stage. According to the Figure 2, there is a several data pre-processing task including Case Folding, Punctuation Removal, Whitespace Removal, Numbers Removal, Stopword Removal, Lemmatization. Case folding is the process to convert all input words into the same form, for instance uppercase or lowercase [15]. So, we transform all our text in description product as the features to lowercase. After that, our text data must be clean from punctuation marks and symbol, so we applied punctuation removal. Next, we applied whitespace removal to remove an unpredicted extra spaces between every word and line or paragraph spacing [16]. We must make sure that our texts only contain meaningful words which aim to represent the essence of each text. So, we need to apply stopword removal. Stopwords are actually the most common words in any language that appears too much in a text does not add much information, such as articles, prepositions, pronouns and conjunctions. Final step in text preprocessing is lemmatization. Lemmatization works to reduce a word variant to its lemma and uses vocabulary and morphological analysis for returning words to their dictionary form [17]. This step converts all of word in our texts to its basic form. Generally, lemmatazation and stemming is a similar approach and often produce same results, but sometimes the basic form of the word may be different than the stemming approach e.g. "caring" is stemmed to "car", but in lemmatization you will get "care" which more appropriate than stemming.Also, in Boban [17] study, Lemmatization produces better results.

2.4.1. Create data tensor

After our text successfully passing data preprocessing stage. We need to vectorize our features by convert our text data into either a sequence of integers and mapping it into real-valued vector, so we can feed it through input layer in our deep neural network models. Also, we limit the total number of words in our text features to the most frequent words, and zero out the rest. We determine the maximum sentence length (number of words) in each text features that will truncating long reviews and pad the shorter reviews with zero values in the next process. According to Figure 2 there are some steps in converting our text data after lemmatizing step called creat data tensor. First, we use tokenizer to split each word in the text. Second, we create an index-based dictionary on each word based on the text we have or the description of skincare products. Next, we transform our tokens from first step into sequence of integer based from our index-based dictionary. Then, truncate and pad the input sequences, so they are all in the same length for modeling. Last step is converting our categorical labels to numbers.

2.5. Model building and training

Next stages are model building and training. Before that, we split our dataset into three part for training, testing, and validating. We build our LSTM and Bi-LSTM model with a similar layer structur as illustrated in Figure 3.

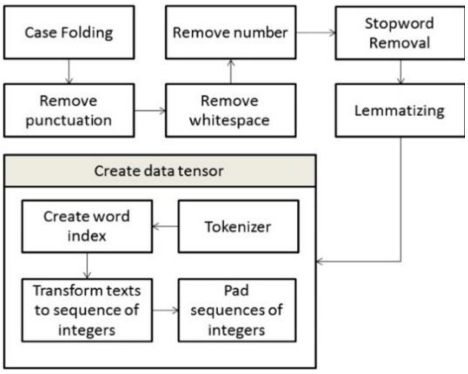


Figure 2. Text preprocessing

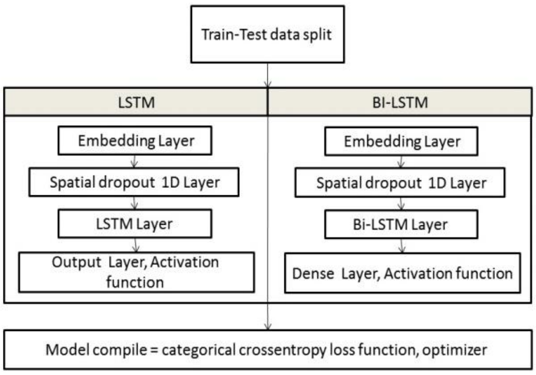


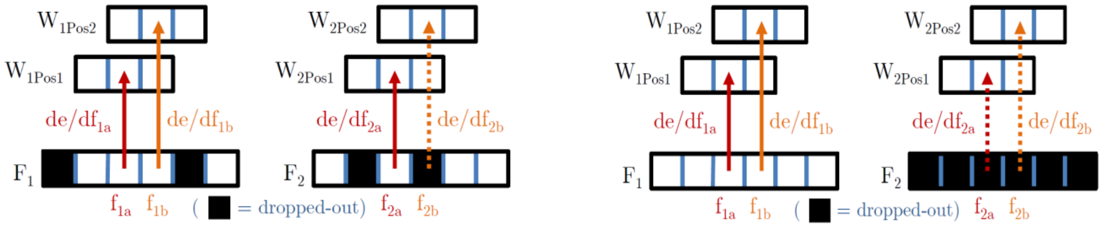
Figure 3. Model architecture

2.5.1. Embedding layer

We put Embedding layer in first place as input layer ad map each word into a real-valued vector to represent each word. Embedding layers works by mapping a raw user/items features in a high dimensional space to dense vectors in a low dimensional embedding space [18]. Basically, embedding layer has similar purpose as popular word embedding frameworks (e.g word2vec and gloVe) which provide a dense representation of words and their relative meanings. However, there is a different between them, which are their training process. Popular word embeddings framework like word2vec and gloVe is trained to predict if word belongs to the context, given other words, e.g., to tell if "cuisine" is a likely word given the "The chef is making a chinese ... " sentence begging. Word2vec learns that "chef" is something that is likely to appear together with "cuisine", but also with "worker", or "restaurant", so it is somehow similar to "waitress", so word2vec learn something about the language. The conclusion is embeddings created by word2vec, gloVe, or other similar frameworks learn to represent words with similar meanings using similar vectors. Meanwhile, embeddings learned from layer of neural network may be trained to predict a specific case, in this case is text classification. So, the embeddings would learn features that are relevant for our text classification. If word2vec has a pre-trained corpus or dictionary, otherwise, embedding layers doesn’t have it. But we already created the index-based dictionary on each word from our features before and transform our features to sequence of integer through it. It’s more efficient, doesn’t need high computing resources, and useful for classification than using pre-trained word embedding like word2vec, even though embedding layer doesn’t capture the semantic similarity of words like word2vec does [19].

2.5.2. Spatial dropout 1D layer

Next, we adding spatial dropout 1D layer. This layer performs the same function as dropout. In standard dropout, the neuron on neural network drops independently as shown in Figure 4(a) [20]. Meanwhile, in spatial dropout it drops entire 1D feature maps instead of individual elements as shown in Figure 4(b).



(a) (b)

Figure 4. Shows the difference between, (a) regular dropout that drops neuron in neural network independently and (b) spatial dropout 1D that drops entire 1D feature maps

2.5.3. Unidirectional and bidirectional long short-term memory (Bi-LSTM)

Next, we use the LSTM layer and the Bi-LSTM layer on each of the two architectural models created. LSTM is very popular for dealing with cases such as NLP, video, and audio where the data is in the form of a sequence. When compared with its predecessor vanilla RNN algorithm which is unable to use past information, LSTM outperforms it with its long-term memory. LSTM transforms the memory shape of cells withinside the RNN via way of means of reworking the tanh activation characteristic layer withinside the RNN right into a shape containing memory devices and gate mechanisms, pursuits to determine how to make use of and replace data saved in memory cells [21]. Now, there is a new concept of mechanism in those sequence feed-forward neural network which called bidirectional. Bidirectional is a mechanism that able to make a neural network works like two-way mirror, which trains an input data twice through past and future. With implementing the bidirectional concept, a regular LSTM not only capable train the input data forward, but also backward. According to Figure 5, Figures 6(a) and 6(b), those models are used the following formula to calculate the predict values,

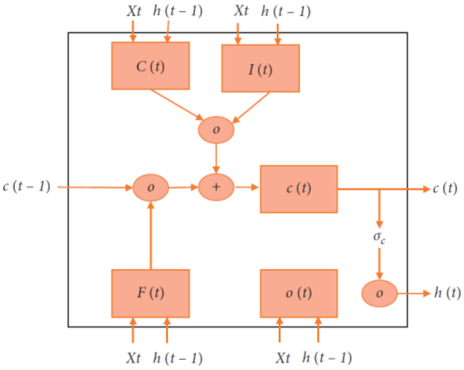
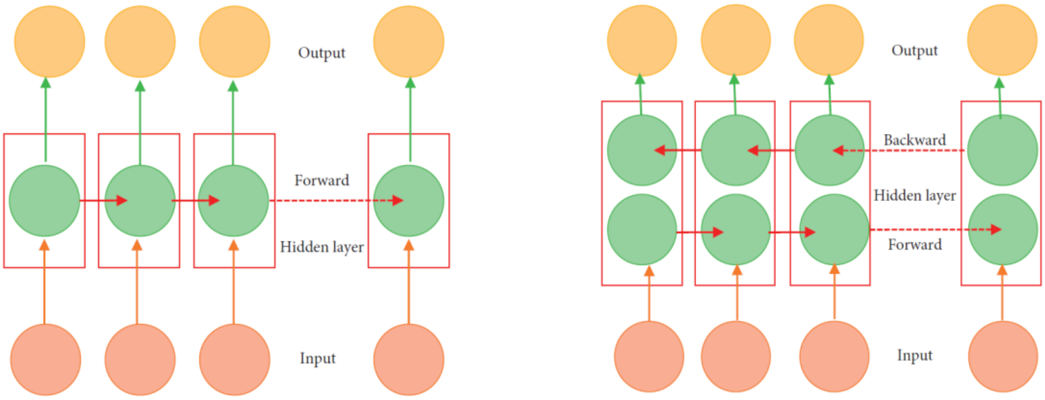


Figure 5. LSTM architecture



(a)

(b)

Figure 6. Shows the differences of LSTM in: (a) unidirectional and (b) Bidirectional mechanism

$$\begin{aligned} &l_t(\text{Input Gate}) = \sigma_g(W_i X_t + R_i h_{t-1} + b_i), \\ &f_t(\text{Forget Gate}) = \sigma_g\left(W_f X_t + R_f h_{t-1} + b_f\right), \\ &Ct(\text{Cell Candidate}) = \sigma_g(W_c X_t + R_c h_{t-1} + b_c), \\ &Ot(\text{Output Gate}) = \sigma_g(W_o X_t + R_o h_{t-1} + b_o), \end{aligned} \tag{1}$$

σ_g = The gate activation function
W_i, W_f, W_c, and W_o = Input weight matrices
R_i, R_f, R_c, and R_o = Recurrent weight matrices
X_t = The data input.
h_{t-1} = The output at the previous time (t − 1)
b_i, b_f, b_c, and b_o = The bias vector

the forget gate counts the measure that decide to removes the previous memory values from the cell state. Just like the forget gate, the input gate determines the new input to the cell state. Then, the LSTM's cell state Ct and the output Ht at time *t* are calculated,

$$\begin{aligned} C_t &= f_t \odot C_t \odot 1 + l_t \odot g_t \\ H_t &= O_t \odot \sigma c(C_t) \end{aligned} \tag{2}$$

$\odot$ = denotes the Hadamard product (element-wise multiplication of vectors)

Also, we use another parameter in our hidden layer and output layer of both LSTM and Bi-LSTM which are dropout, recurrent dropout, recurrent regularizer, L2 regularizers. Recurrent dropout is a regularization that devoted recurrent neural network algorithms. Recurrent dropout works differently from the usual dropout, which is applied to for-ward connections of feed-forward architectures or RNNs, drop neurons directly in recurrent connections in away that does not cause loss of long-term memory instead [22]. There is a formula update on Ct when implementing recurrent dropout to the cell update vector gt,

$$C_t = f_t \odot C_t \odot 1 + l_t \odot d(g_t) \tag{3}$$

Where d is dropout. Next parameter is usual dropout that we apply same with recurrent dropout where in both LSTM and Bi-LSTM layer. Last parameter is L2 regularizers which is a layer weight regularizers that enforce penalties on layer parameters or layer activity during optimization process. These penalties are added up in a loss function that optimizes the network applied on a per-layer basis there are three ways to apply these regularizer, in layer's kernel, bias, and output. L2 regularizer summed the suared weights to the loss function. L2 are often to set a value on logarithmic scale between 0 and 0.1, such as 0.1, 0.001 and 0.0001.

2.6. Model evaluation and prediction

Final stages are model evaluation and prediction with a validation dataset. The evaluation contains a several score to measure the performance of model training and testing. We use an accuracy score by obtaining precision, recall, and f-measure.

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN} \tag{4}$$

TP = True Positive is a skin concern that is in the actual label and appears in the prediction.
FP =False Poisitive is a skin concern that is in actual label but doesn't appears in the prediction.
FN = False Negative is skin concern that is not in the actual label but appears in the prediction.
TN = True Negative is a skin concern that is neither in the actual label nor the prediction

Precision is the percentage of positive cases that were actually predicted to be truly positive [23]. Precision is calculated,

$$\text{Precision} = \frac{TP}{TP+FP} \tag{5}$$

recall is the Percentage of actual positive cases that were correctly predicted. It actually measures the coverage of positive cases and accurately reflects the predicted cases [23]. Recall is calculated,

$$\text{Recall} = \frac{TP}{TP+FN} \tag{6}$$

F1- Measure is a composite measure that captures the trade-offs related to precision and recall and calculated,

$$\textit{F1-Measure} = \frac{Precision\ xRecall}{Precision+Recall} \tag{7}$$

loss function that being used is categorical cross-entropy. Categorical cross-entropy is specifically used for the case of multi-class classification which increasing or decreasing the relative penalty of a probabilistic false negative for an individual class [24]. The categorical cross-entropy loss function is used the following formula,

$$\text{Loss} = -\sum_{i=1}^{\substack{\text{output}\\ \text{size}}} y_i \cdot \log \hat{y}_i \tag{8}$$

$\hat{y}_i$ = i-th scalar value in the model output
y_i = Corresponding target value
Output Size = The number of scalar values in the model output
this loss function measures the distance of dissimilarity between the true label distribution and the predicted label distribution. The y_i defines the probability that event i occurs. The sum of all y_i is 1 that means one event may occur. The minus sign guarantees that the closer the distributions are to each other, the smaller the loss. Also, we use a confusion matrix to calculate the total of true or false a prediction generated by the classification model. Confusion matrix is machine learning concept that contains information about the actual and predicted classifications performed by the classification system which has two dimention divided for indexing the actual class of an object, and the other is indexing the class that the classifier predicts [25].

3. RESULTS AND DISCUSSION

All stages of this research were carried out with the python programming language. The results of this research are measured using several scores. The scores measure the performance of the proposed model classification prediction, by looking at the accuracy and loss scores in each experiment carried out.

3.1. Train-test-validation split evaluation

The first experiment was carried out by splitting the dataset into three parts. The three parts of dataset used for train, test, and validation process. Table 1 shows the result of dataset spliting with the best result of 80% train dataset, 1% test dataset, and 19% validation dataset with an acuracy score of 98.04% and loss 19.19% from Bi-LSTM.

Table 1. The result based on the distribution of dataset splitting

Train/Test/Validation Split	LSTM Accuracy	LSTM Loss	Bi-LSTM Accuracy	Bi-LSTM Loss
80/1/19	0.9412	0.1991	0.9804	0.1919
80/ 2/18	0.9401	0.2069	0.9800	0.1991
80/3/17	0.9405	0.1919	0.9801	0.2007
80/4/16	0.9400	0.2020	0.9611	0.1910
80/5/15	0.9258	0.2276	0.9690	0.2205
80/6/14	0.9635	0.2481	0.9800	0.2287
80/7/13	0.9643	0.2311	0.9750	0.2210
80/8/12	0.9681	0.2490	0.9800	0.2294
80/9/11	0.9292	0.2410	0.9790	0.2910
80/10/10	0.9261	0.2911	0.9601	0.2411
80/11/9	0.9600	0.2450	0.9780	0.2910
80/12/8	0.9278	0.2101	0.9501	0.2934
80/13/7	0.9210	0.2105	0.9309	0.2451
80/14/6	0.9181	0.2980	0.9187	0.2410
80/15/5	0.9082	0.2949	0.9182	0.2410
80/16/4	0.9009	0.2910	0.8890	0.3800
80/17/3	0.8829	0.3991	0.8898	0.3809
80/18/2	0.8821	0.3929	0.8810	0.3876
80/19/1	0.8832	0.3901	0.8824	0.3792
90/1/9	0.9290	0.3519	0.9790	0.2509
90/2/8	0.9283	0.3210	0.9174	0.2410
90/3/7	0.9043	0.3210	0.9111	0.2901
90/4/6	0.9021	0.3410	0.9019.	0.3100
90/5/5	0.9080	0.3410	0.8978	0.3240
90/6/5	0.8880	0.3450	0.8901	0.3210
90/7/3	0.8821	0.3421	0.8981	0.3209
90/8/2	0.8901	0.3450	0.8999	0.3210
90/9/1	0.9059	0.3592	0.9079	0.3465
Best Score			0.9804	0.1919

3.2. Hyper-parameters tuning

There are a several hyper-parameters used in model training. Memory units (Mu), Optimizers (O), Activity function (Af) tuning as shown in Table 2. The Bi-LSTM model still outperformed the LSTM with a memory unit setting of 100, RMSprop optimizers, and Activity function softmax.

Next, early stopping callback is a parameter that stop the training process when metric has stopped improving by stores the model’s weights at the optimal epoch. These parameter attain the highest accuracy intraining regardless of the epoch setting [26]. These parameter has two hyper-parameter which is patience (p) and minimal delta (-Δ). The result of tuning these two hyper-parameter as shown in Table 3. The Bi-LSTM model still outperformed the LSTM with a patience of 5 and min delta of 0.0001.

Table 2. Memory units, optimizers, activity function tuning

Model	Mu/O	Mu = 100	Mu = 200	Af
LSTM	Adam	Acc=0.9054 Loss=0.3811	Acc=0.8899 Loss=0.3723	Softmax
	RMSprop	Acc=0.9412 Loss=0.1991	Acc=0.9009 Loss=0.3792	
	SGD	Acc=0.7821 Loss=0.5978	Acc=0.7811 Loss=0.5985	
	Adadelta	Acc= 0.7890 Loss=0.5821	Acc= 0.7799 Loss=0.4951	
	Adagrad	Acc=0.7829 Loss=0.5435	Acc=0.7826 Loss=0.5433	
	Adam	Acc=0.8054 Loss=0.4811	Acc=0.8099 Loss=0.4723	Sigmoid
	RMSprop	Acc=0.8059 Loss=0.4592	Acc=0.7963 Loss=0.4352	
	SGD	Acc=0.6821 Loss=0.6978	Acc=0.6816 Loss=0.6985	
	Adadelta	Acc= 0.6890 Loss=0.6821	Acc= 0.6799 Loss=0.5951	
	Adagrad	Acc=0.7829 Loss=0.6435	Acc=0.6826 Loss=0.6433	
	Adam	Acc=0.5063 Loss=0.5841	Acc=0.5099 Loss=0.4323	ReLu
	RMSprop	Acc=0.5059 Loss=0.5592	Acc=0.5003 Loss=0.5365	
	SGD	Acc=0.6821 Loss=0.6978	Acc=0.4821 Loss=0.5953	
	Adadelta	Acc= 0.5890 Loss=0.5821	Acc= 0.5939 Loss=0.5951	
	Adagrad	Acc=0.6829 Loss=0.5435	Acc=0.5826 Loss=0.6433	
	Adam	Acc=0.5054 Loss=0.5811	Acc=0.5099 Loss=0.4723	Tanh
	RMSprop	Acc=0.5059 Loss=0.5592	Acc=0.7963 Loss=0.4152	
	SGD	Acc=0.5821 Loss=0.5978	Acc=0.4816 Loss=0.5985	
	Adadelta	Acc= 0.6890 Loss=0.6821	Acc= 0.6799 Loss=0.4951	
	Adagrad	Acc=0.5829 Loss=0.6435	Acc=0.5826 Loss=0.6433	
	Adam	Acc=0.6054 Loss=0.4811	Acc=0.5099 Loss=0.4723	Hard Sigmoid
	RMSprop	Acc=0.6059 Loss=0.4592	Acc=0.7963 Loss=0.4152	
	SGD	Acc=0.6821 Loss=0.4978	Acc=0.6821 Loss=0.4953	
	Adadelta	Acc= 0.6890 Loss=0.4821	Acc= 0.4939 Loss=0.5951	
	Adagrad	Acc=0.6829 Loss=0.4435	Acc=0.6826 Loss=0.4433	
Bi-LSTM	Adam	Acc=0.9034 Loss=0.3851	Acc=0.8889 Loss=0.3783	Softmax
	RMSprop	Acc=0.9804 Loss=0.1919	Acc=0.9069 Loss=0.3592	
	SGD	Acc=0.7861 Loss=0.5988	Acc=0.7881 Loss=0.5985	
	Adadelta	Acc= 0.7890 Loss=0.5871	Acc= 0.7799 Loss=0.4961	
	Adagrad	Acc=0.7869 Loss=0.5495	Acc=0.7876 Loss=0.5483	
	Adam	Acc=0.8054 Loss=0.4811	Acc=0.8089 Loss=0.4763	Sigmoid
	RMSprop	Acc=0.8059 Loss=0.4592	Acc=0.7983 Loss=0.4392	
	SGD	Acc=0.6821 Loss=0.6978	Acc=0.6886 Loss=0.6975	
	Adadelta	Acc= 0.6890 Loss=0.6821	Acc= 0.6799 Loss=0.5671	
	Adagrad	Acc=0.7829 Loss=0.6485	Acc=0.6876 Loss=0.6483	
	Adam	Acc=0.5073 Loss=0.5881	Acc=0.5079 Loss=0.4383	ReLu
	RMSprop	Acc=0.5079 Loss=0.5582	Acc=0.5093 Loss=0.5375	
	SGD	Acc=0.6861 Loss=0.6678	Acc=0.4881 Loss=0.5993	
	Adadelta	Acc= 0.5870 Loss=0.5851	Acc= 0.5989 Loss=0.5971	
	Adagrad	Acc=0.6869 Loss=0.5475	Acc=0.5826 Loss=0.6493	
	Adam	Acc=0.5064 Loss=0.5871	Acc=0.5079 Loss=0.4783	Tanh
	RMSprop	Acc=0.5089 Loss=0.5572	Acc=0.7973 Loss=0.4182	
	SGD	Acc=0.5861 Loss=0.5978	Acc=0.4896 Loss=0.5995	
	Adadelta	Acc= 0.6860 Loss=0.6881	Acc= 0.6779 Loss=0.4971	
	Adagrad	Acc=0.5879 Loss=0.6465	Acc=0.5886 Loss=0.6493	
	Adam	Acc=0.6074 Loss=0.4861	Acc=0.5069 Loss=0.4783	Hard Sigmoid
	RMSprop	Acc=0.6089 Loss=0.4572	Acc=0.7993 Loss=0.4172	
	SGD	Acc=0.6861 Loss=0.4978	Acc=0.6881 Loss=0.4963	
	Adadelta	Acc= 0.6860 Loss=0.4881	Acc= 0.4979 Loss=0.5981	
	Adagrad	Acc=0.6889 Loss=0.4475	Acc=0.6896 Loss=0.4463	

Table 3. Patience and min delta tuning

Model	P/-Δ	-Δ = 0.01	-Δ = 0.001	-Δ = 0.0001
LSTM	P = 1	Acc=0.8999 Loss=0.3492	Acc=0.8854 Loss=0.3111	Acc=0.8814 Loss=0.3121
	P = 2	Acc=0.8829 Loss=0.3461	Acc=0.8839 Loss=0.3413	Acc=0.8889 Loss=0.3433
	P = 3	Acc=0.8899 Loss=0.3401	Acc=0.8808 Loss=0.3323	Acc=0.8878 Loss=0.3333
	P = 4	Acc=0.8959 Loss=0.3392	Acc=0.8854 Loss=0.3111	Acc=0.8821 Loss=0.3111
	P = 5	Acc=0.8829 Loss=0.3400	Acc=0.8808 Loss=0.3323	Acc=0.9412 Loss=0.1991
BI-LSTM	P = 1	Acc=0.9009 Loss=0.3292	Acc=0.8999 Loss=0.3811	Acc=0.8954 Loss=0.3221
	P = 2	Acc=0.8829 Loss=0.3461	Acc=0.8839 Loss=0.3453	Acc=0.8839 Loss=0.3443
	P = 3	Acc=0.8979 Loss=0.3398	Acc=0.8854 Loss=0.3117	Acc=0.8821 Loss=0.3111
	P = 4	Acc=0.8829 Loss=0.3400	Acc=0.8808 Loss=0.3323	Acc=0.9079 Loss=0.3562
	P = 5	Acc=0.8909 Loss=0.3409	Acc=0.8858 Loss=0.3111	Acc=0.9804 Loss=0.1919

3.3. Model evaluation and prediction

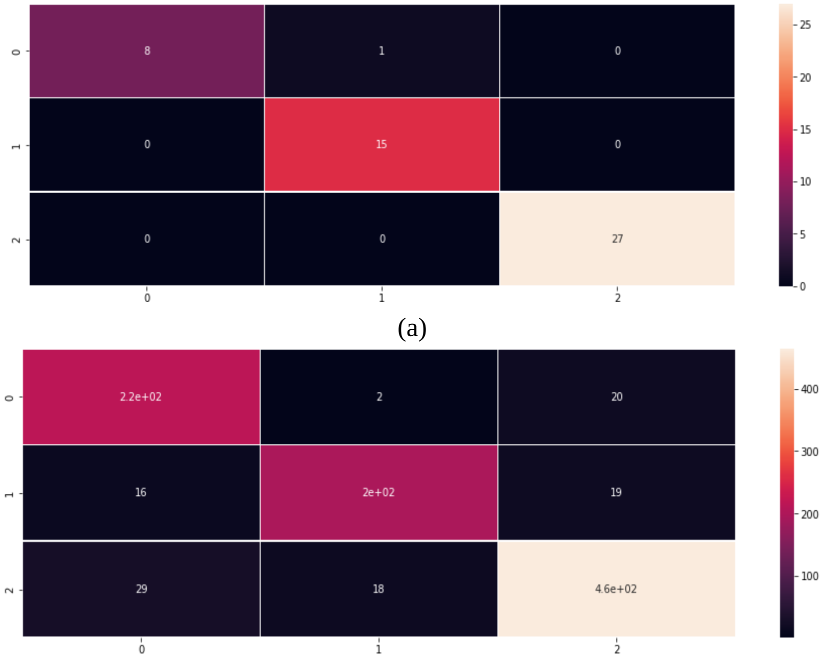
After the hyper-parameter tuning, we get the best settings are as shown in Table 4. We evaluate our proposed models with a validation dataset as much as 980 skincare products. To measure theperformance of model training and testing, we used an accuracy score by obtaining precision, recall, and f-measure as shown in Table 5. Testing and validation confusion matrix in Bi-LSTM models are shown in Figures 7(a) and 7(b).

Table 4. The best models' settings

Hyper-Parameter	LSTM	Bi-LSTM
Train/Test/Validation data split	80/1/9	80/1/9
Max.Number of Words	50000	50000
Max.Sequence Length	512	512
Embedding Dimension	500	500
Memory Units	100	100
Optimizers	RMSprop	RMSprop
Activation Function	Softmax	Softmax
Spatial Dropout 1D	0.3	0.3
Dropout	0.3	0.3
Recurrent Dropout	0.3	0.3
Recurrent Regularizer	0.01	0.01
Kernel Regularizer	0.01	0.01
Bias Regularizer	0.01	0.01
Patience	5	5
Min.Delta	0.0001	0.0001
Accuracy Score =	0.9412	0.9804
Loss Score =	0.1991	0.1919

Table 5. Classification report on the validation data in the proposed models

Model	Precision	Recall	F1-Score	
LSTM	0.9012	0.8939	0.8975	Micro Avg
	0.9004	0.8797	0.8894	Macro Avg
	0.9015	0.8939	0.8972	Weighted Avg
	0.8939	0.9939	0.8939	Samples Avg
BI-LSTM	0.8981	0.8908	0.8945	Micro Avg
	0.8905	0.8839	0.8866	Macro Avg
	0.8994	0.8908	0.8946	Weighted Avg
	0.8908	0.8908	0.8908	Samples Avg



(a)

(b)

Figure 7. BI-LSTM Testing, (a) validation and (b) confusion matrix

3.3.1. Models inference

After getting the fine-tuned in each model, we tested the models to predicting what's skin concern that every skincare product overcomes by manually input the skincare product description to the models. The actual labels over skincare description that we manually input before are taken from official website of each skincare products. The results can be seen in Table 6.

Table 6. Models inference

Skincare Description	Actual Label	LSTM Prediction	Bi-LSTM Prediction
Niacinamide 10% + Zinc 1% from The Ordinary is a water-based vitamin and mineral formula with 10% niacinamide and 1% zinc PCA. This water-based serum is great for those looking for solutions for visible shine/enlarged pores/textural irregularities Benefits	acne, big pores, blemish	acne, big pores, blemish	acne, big pores, blemish
Address signs of ageing with the Retinol Serum 0.2% in Squalane from The Ordinary; a water-free, multipurpose, potent solution formulated to refine pores, reduce the appearance of dark spots and wrinkles and improve skin texture. Enriched with a 0.2% concentration of the anti-ageing powerhouse Retinol, which is a derivative of Vitamin A, the lightweight serum has a plumping and firming effect on the complexion, as well as protecting the skin from harmful environmental aggressors. Another key antioxidant ingredient Squalane prevents UV damage and the formation of age spots whilst counteracting harmful bacteria, leaving you with flawless skin.	anti-aging, wrinkles	anti-aging, wrinkles	anti-aging, wrinkles
Quench your skin in a wave of pure hydration with The INKEY List Hyaluronic Acid Serum. This powerful ingredient attracts up to 1000x its weight in water, binding moisture to restore the skin’s natural barrier. The gentle serum is suitable for all skin types to restore balance.	drynes, redness	drynes, redness	drynes, redness

4. CONCLUSION

The findings have produced a satisfactory performance, with an adequate score obtained both for accuracy and loss. The performance of the bidirectional LSTM model, which makes use of this bidirectional mechanism, outperforms that of the LSTM model, which makes use of the same mechanism and produces an accuracy score of 94.12% and a loss value of 19.91%. The bidirectional LSTM model produces a score of 98.04% for its accuracy and a loss value of 19.19% for its loss. The usage of an embedding layer where the data was previously transformed into a tensor form may be adjusted by employing a popular word embedding such as word2vec or gloVe, which requires a large amount of computer resources but can extract the semantic meaning of the features. Both of the models that have been provided have been able to effectively train on the dataset that we obtain from well-known websites that specialize in the sale of skincare items. Because of this, the prediction successfully maps the skin's worries over the description of each skincare product, both with unseen data or validation data and the description that we manually enter into the models. Additionally, given the dataset that we have, this research has the potential to be further expanded into a recommendation system for online retailers that offer skincare goods as well as a mobile application.

ACKNOWLEDGEMENTS

We would like to thank all colleagues at the Faculty of Computer Science, Universitas Mercu Buana who were involved in this research, either in terms of knowledge assistance and for their other support.

REFERENCES

[1] J. E. Lee, M. L. Goh, and M. N. Bin Mohd Noor, “Understanding purchase intention of university students towards skin care products,” *PSU Research Review*, vol. 3, no. 3, pp. 161–178, 2019, doi: 10.1108/prr-11-2018-0031.

[2] H. Symum, F. Islam, H. K. Hiya, and K. M. A. Sagor, “Assessment of the Impact of COVID-19 pandemic on population level interest in Skincare: Evidence from a google trends-based Infodemiology study,” *medRxiv*, 2020, doi: 10.1101/2020.11.16.20232868.

[3] Indriyani and I. Made Sudarma, “Classification of facial skin type using discrete wavelet transform, contrast, local binary pattern and support vector machine,” *Journal of Theoretical and Applied Information Technology*, vol. 98, no. 5, pp. 768–779, 2020.

[4] A. Borrego-Sánchez, C. I. Sainz-Díaz, L. Perioli, and C. Viseras, “Theoretical study of retinol, niacinamide and glycolic acid with halloysite clay mineral as active ingredients for topical skin care formulations,” *Molecules*, vol. 26, no. 15, 2021, doi: 10.3390/molecules26154392.

[5] S. Khezri and K. Khezri, “The side effects of cosmetic consumption and personal care products,” *Journal of Advanced Chemical and Pharmaceutical Materials (JACPM)*, vol. 2, no. 3, pp. 152–156, 2019, [Online]. Available: http://advchempharm.ir/journal/index.php/JACPM/article/view/121%0Ahttp://advchempharm.ir/journal/index.php/JACPM/article/download/121/258.

[6] S. Cho *et al.*, “Knowledge and behavior regarding cosmetics in Koreans visiting dermatology clinics,” *Annals of Dermatology*, vol. 29, no. 2, pp. 180–186, 2017, doi: 10.5021/ad.2017.29.2.180.

[7] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, “Deep Learning Based Text Classification: A Comprehensive Review,” 2020, [Online]. Available: http://arxiv.org/abs/2004.03705.

[8] A. Arami, A. Poulakakis-Daktylidis, Y. F. Tai, and E. Burdet, “Prediction of Gait Freezing in Parkinsonian Patients: A Binary Classification Augmented With Time Series Prediction,” *IEEE transactions on neural systems and rehabilitation engineering : a publication of the IEEE Engineering in Medicine and Biology Society*, vol. 27, no. 9, pp. 1909–1919, 2019, doi: 10.1109/TNSRE.2019.2933626.

[9] L. Tang, Y. Tian, and P. M. Pardalos, “A novel perspective on multiclass classification: Regular simplex support vector machine,” *Information Sciences*, vol. 480, pp. 324–338, 2019, doi: 10.1016/j.ins.2018.12.026.

[10] S. A. Wulandari, W. A. Prasetyanto, and M. D. Kurniatie, “Classification of Normal , Oily and Dry Skin Types Using a 4-

Connectivity and 8-Connectivity Region Properties Based on Average Characteristics of Bound," *Jurnal Transformatika*, vol. 17, no. 01, pp. 78–87, 2019, [Online]. Available: journals.usm.ac.id/index.php/transformatika.

[11] J. Zheng and L. Zheng, "A Hybrid Bidirectional Recurrent Convolutional Neural Network Attention-Based Model for Text Classification," *IEEE Access*, vol. 7, pp. 106673–106685, 2019, doi: 10.1109/ACCESS.2019.2932619.

[12] R. L. Abduljabbar, H. Dia, and P. W. Tsai, "Unidirectional and bidirectional LSTM models for short-term traffic prediction," *Journal of Advanced Transportation*, vol. 2021, 2021, doi: 10.1155/2021/5589075.

[13] R. Diouf, E. N. Sarr, O. Sall, B. Birregah, M. Bousso, and S. N. Mbaye, "Web Scraping: State-of-the-Art and Areas of Application," *Proceedings - 2019 IEEE International Conference on Big Data, Big Data 2019*, pp. 6040–6042, 2019, doi: 10.1109/BigData47090.2019.9005594.

[14] S. Hara, A. Nitanda, and T. Maehara, "Data cleansing for models trained with SGD," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[15] M. A. Rosid, A. S. Fitrani, I. R. I. Astutik, N. I. Mulloh, and H. A. Gozali, "Improving Text Preprocessing for Student Complaint Document Classification Using Sastrawi," *IOP Conference Series: Materials Science and Engineering*, vol. 874, no. 1, 2020, doi: 10.1088/1757-899X/874/1/012017.

[16] F. Mohammad, "Is preprocessing of text really worth your time for toxic comment classification?," *2018 World Congress in Computer Science, Computer Engineering and Applied Computing, CSCE 2018 - Proceedings of the 2018 International Conference on Artificial Intelligence, ICAI 2018*, pp. 447–453, 2018.

[17] I. Boban, A. Doko, and S. Gotovac, "Sentence retrieval using Stemming and Lemmatization with different length of the queries," *Advances in Science, Technology and Engineering Systems*, vol. 5, no. 3, pp. 349–354, 2020, doi: 10.25046/aj050345.

[18] X. Zhaok *et al.*, "AutoEmb: Automated Embedding Dimensionality Search in Streaming Recommendations," *Proceedings - IEEE International Conference on Data Mining, ICDM*, vol. 2021-December, pp. 896–905, 2021, doi: 10.1109/ICDM51629.2021.00101.

[19] D. López-Sánchez, J. R. Herrero, A. G. Arrieta, and J. M. Corchado, "Hybridizing metric learning and case-based reasoning for adaptable clickbait detection," *Applied Intelligence*, vol. 48, no. 9, pp. 2967–2982, 2018, doi: 10.1007/s10489-017-1109-7.

[20] G. Cheng, V. Peddinti, D. Povey, V. Manohar, S. Khudanpur, and Y. Yan, "An exploration of dropout with LSTMs," *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH*, vol. 2017-August, pp. 1586–1590, 2017, doi: 10.21437/Interspeech.2017-129.

[21] D. Fitrianah and R. N. Jauhari, "Extractive text summarization for scientific journal articles using long short-term memory and gated recurrent units," *Bulletin of Electrical Engineering and Informatics*, vol. 11, no. 1, pp. 150–157, 2022, doi: 10.11591/eei.v11i1.3278.

[22] A. Dutta, S. Kumar, and M. Basu, "A Gated Recurrent Unit Approach to Bitcoin Price Prediction," *Journal of Risk and Financial Management*, vol. 13, no. 2, p. 23, 2020, doi: 10.3390/jrfm13020023.

[23] F. Ramzan *et al.*, "A Deep Learning Approach for Automated Diagnosis and Multi-Class Classification of Alzheimer's Disease Stages Using Resting-State fMRI and Residual Neural Networks," *Journal of Medical Systems*, vol. 44, no. 2, 2020, doi: 10.1007/s10916-019-1475-2.

[24] Y. Ho and S. Wookey, "The Real-World-Weight Cross-Entropy Loss Function: Modeling the Costs of Mislabeling," *IEEE Access*, vol. 8, pp. 4806–4813, 2020, doi: 10.1109/ACCESS.2019.2962617.

[25] J. Xu, Y. Zhang, and D. Miao, "Three-way confusion matrix for classification: A measure driven view," *Information Sciences*, vol. 507, pp. 772–794, 2020, doi: 10.1016/j.ins.2019.06.064.

[26] S. Shin, Y. Lee, M. Kim, J. Park, S. Lee, and K. Min, "Deep neural network model with Bayesian hyperparameter optimization for prediction of NOx at transient conditions in a diesel engine," *Engineering Applications of Artificial Intelligence*, vol. 94, 2020, doi: 10.1016/j.engappai.2020.103761.

BIOGRAPHIES OF AUTHORS

Devi Fitrianah is working in Bina Nusantara University as a lecturer in Department of Computer Science, School of Computer Science. Her academic qualification is Dr., S.Kom., M.T.I. As Data Scientist, she does research in Artificial Intelligence, Data Mining, Machine Learning and Information Systems (Business Informatics). Recent researches are in the field of Natural Language Processing for sentiment analysis and Automated Text Summarization. She can be contacted at email: devifitrianah1@gmail.com.

Andre Hangga Wangsa is Student in Universitas Mercu Buana who are pursuing a bachelor's degree in Computer Science His interest are in Data Science fields such as Natural Language Processing and Computer Vision. He can be contacted at email: andrehanggaw@gmail.com.