Materials Today: Proceedings 80 (2023) 3730–3743

Contents lists available at ScienceDirect

Materials Today: Proceedings

journal homepage: www.elsevier.com/locate/matpr

An Efficient Approach of Product Recommendation System using NLP Technique

Akhilesh Kumar Sharma [a], Bhavna Bajpai [b,*], Rachit Adhvaryu [c], Suthar Dhruvi Pankajkumar [d], Prajapati Parthkumar Gordhanbhai [e], Atul Kumar [f]

[a] Department Of Information Technology, Manipal University Jaipur, Rajasthan, India
[b] Dr C V Raman University, Khandwa, India
[c] Department of Computer Engineering, Marwadi University ,Rajkot, India
[d] Alpha College Of Engineering & Technology Khatraj, Kalol, India
[e] Sankalchand Patel College Of Engineering, Visnagar, India
[f] Dr. D.Y. Patil B-School, Tathawade, Pune, India

ARTICLE INFO

Article history:
Received 8 July 2021
Accepted 17 July 2021
Available online 6 August 2021

Keywords:
NLP
TF-ID
Recommendation System
CNN

ABSTRACT

As we are moving toward an age of digital globalization and online shopping, there is an increasing need for an efficient and reliable system that can help the consumers and the visitors to find their suitable products. Currently, various websites display the searched product when a visitor comes to their website. What we need is a system, which can recommend the products which are like the searched products. This will help the consumer to find out another product in case the item is unavailable, or the searched product is not good enough, or when they would like to look through different similar products. A good recommendation system has been found out to be financially beneficial for the companies also. It is found out that consumer is 35% more likely to buy a product if the recommendation is good enough for consumers.

This proposed approach to the problem of the product recommendation system is to make use of the Amazon Apparel database, which contains data of 180,000 apparels. We are going to use NLP technologies and CNN to help in predicting similar products. Title of the product is used as a major attribute during NLP analysis and recommendation. CNN used at last to create a feature vector from product images, and use this vector combined with all the other vectors, for prediction. We compare the distance between vectors of all products and recommend the products with least distance. VGG-16 architecture used to extract the features from the images.

Selection and peer-review under responsibility of the scientific committee of the International Conference on Nanoelectronics, Nanophotonics, Nanomaterials, Nanobioscience & Nanotechnology.

1. Introduction

A product recommendation system is a software application that gathers and provides recommendations for things or content that a certain user might like to acquire. This approach uses Machine learning algorithms and a variety of data about particular items and users.

We intend to build a recommendation system that can predict products that are available in database and that are like chosen product. We will need to select various features which can be useful in recommending products. We will also keep testing various algorithms and find out the most suitable algorithm to work with. We will also use various combination of vectors to find out the best, which can give us a good result.

1.1. Euclidean distance

In maths, it's a distance function with two points in Euclidean space [1,2].

Formula:

$$d(p,q) = d(q,p) = \sqrt{\left(\sum_{i=1}^{n}(qi-pi)^2\right)} \quad (1)$$

* Corresponding author.
E-mail address: bhavna.bajpai0212@gmail.com (B. Bajpai).

https://doi.org/10.1016/j.matpr.2021.07.371

Selection and peer-review under responsibility of the scientific committee of the International Conference on Nanoelectronics, Nanophotonics, Nanomaterials, Nanobioscience & Nanotechnology.

We indeed use the weighted Euclidean distance between AA and BB

$$d(A,B) = \sqrt{\left(\sum_{i=1}^{n} w_i(A_i - B_i)^2\right)} \quad (2)$$

With the help of weighted Euclidean distance, we can put different importance to different features of our product. This help in having our predictions and recommendations much better and modifiable.

1.1.1. Content based recommendation

In content-based [3,4] suggestion, the system will suggest anything that looks like the item you selected.

In the suggestion stage, the system first analyses similarities between all pairs of things, then uses the most similar items to a user's already-selected items to generate a list of recommendations [5-7] may be explicitly or implicitly. As the customer provides new sources of information or performs actions based on the recommendations, the system becomes more precise [8,9].

1.2. Bag of words

It's a way of showing textual data to mode text along with the ML technique [10,11]. Complexity comes both choose the way to design vocab of the known wordsin Fig. 1, and way to attain the fact of known words.

2 TF-IDF (Term frequency - inverse document frequency)

It is a strategy for quantifying a word in a document by assigning a weight to each word that represents the word's importance in the document and corpus.It's a simple and common technique in text mining and IR [12,13].

TF-IDF = Term Frequency (TF) * Inverse Document Frequency (IDF)

In document set D, it is calculated as follows [14,15].

$$tf(t,d) = log(1 + freq(t,d)) \quad (3)$$

$$idf(t,D) = \log\left(\frac{N}{count(d \epsilon D : t \epsilon d)}\right) \quad (4)$$

1.2.1. Word2Vec

They're vector representations of a word. Word2Vec is one among the foremost popular technique to find out word embeddings using shallow neural network. Consider the following similar sentences: Have an honest day and [16,17] Have an excellent day [23,24]. They hardly have different meaning as in Figs. 2 and 3. In BOW and TF-IDF, they're considered different but in Word2Vec they're treated same.

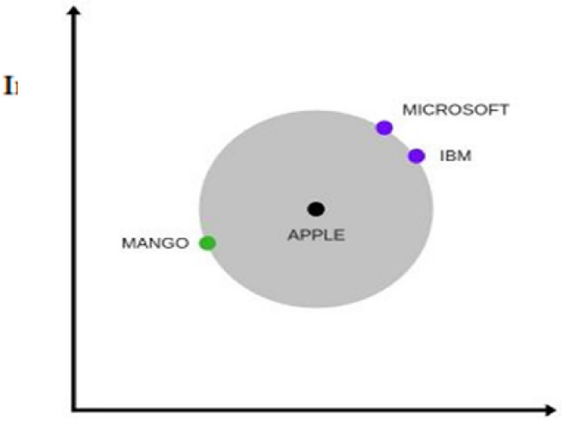


Fig 2. Word2Vec Space Representation [7,8].

1.3. Convolution neural network (CNN)

A Convolutional Neural Network (ConvNet/CNN) is a deep Learning algorithm, take input as image as in Fig. 4, and assign weights to thefeatures [18,19].The image and the processing took place in it [21,22].

1.4. Vgg16

VGG-16 is CNN architecture, with 16 layers. Its layers contain CN layers, Max-pooling, Activation, Fully connected layers as in Fig. 5.

There are 13 convolutional layers, 5 Max Pooling layers and 3 Dense layers which sums up to 21 layers but only 16 wt layers. VGG-16 Fig. 5 neural network is trained on ImageNet dataset which has around 14 million images and 1000 classes and achieves 92.7% in top-5 accuracy [12,13]. Moreover researchers are proposing various protocols in the field of healthcare [25-30] and vehicle communication [31-37] to protect the information exchanged among various devices to devices.

2. Methodology

2.1. Dataset

Our dataset has the data of women's apparel from Amazon website. We contain dataset of around 180,000 clothes. The dataset has 19 columns giving information about each product. Some of these attributes are not useful for us in building a recommender system. We will do data analysis and exploration on each attribute

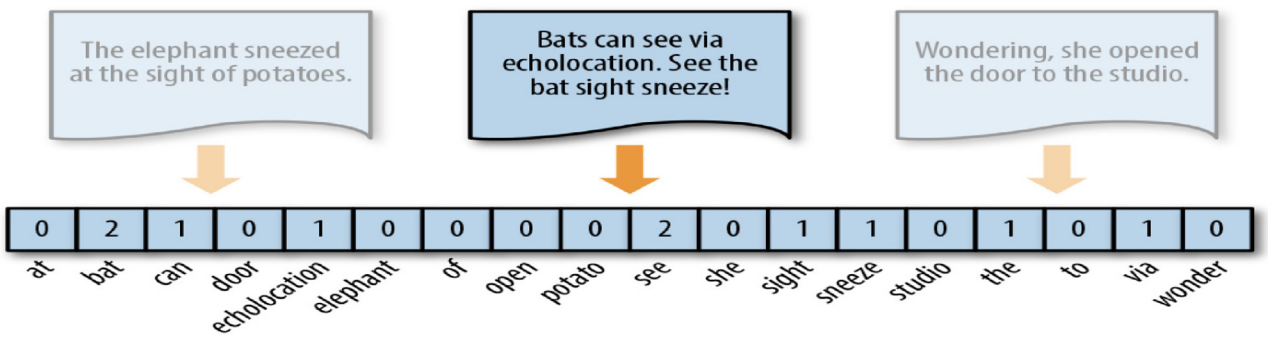


Fig 1. Word Vector with Bow [5].

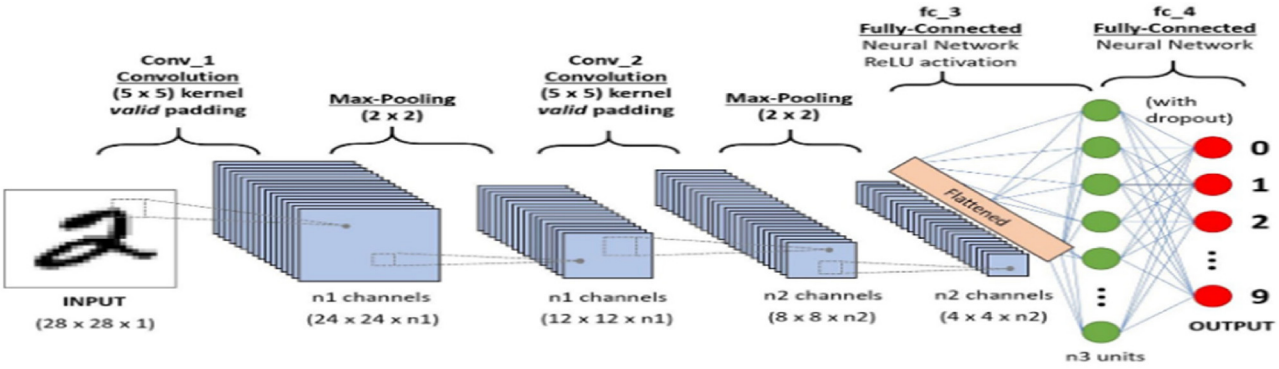


Fig 3. Word2Vec ConvNet [7].

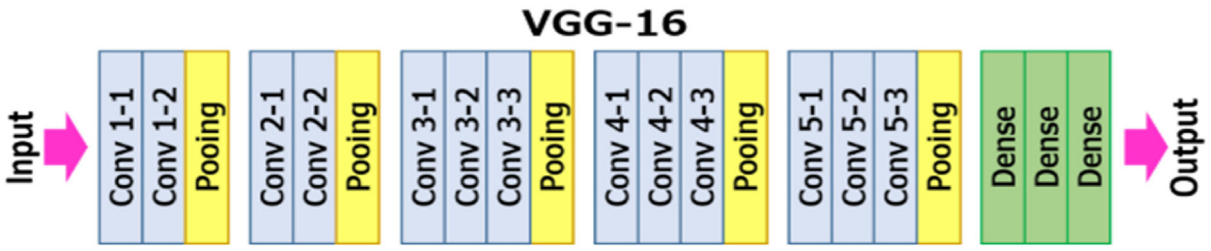


Fig 4. CNN Architecture [10].

	asin	brand	color	medium_image_url	product_type_name	title	formatted_price
0	B016I2TS4W	FNC7C	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	Minions Como Superheroes Ironman Long Sleeve R...	None
1	B01N49AI08	FIG Clothing	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	FIG Clothing Womens Izo Tunic	None
2	B01JDPCOHO	FIG Clothing	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	FIG Clothing Womens Won Top	None
3	B01N19U5H5	Focal18	None	https://images-na.ssl-images-amazon.com/images...	SHIRT	Focal18 Sailor Collar Bubble Sleeve Blouse Shi...	None
4	B004GSI2OS	FeatherLite	Onyx Black/ Stone	https://images-na.ssl-images-amazon.com/images...	SHIRT	Featherlite Ladies' Long Sleeve Stain Resistan...	$26.26

Fig 5. VGG-16 Layers Architecture.

```
def nlp_preprocessing(total_text, index, column):
#     if type(total_text) is not int:
        string = ""
        for words in total_text.split():
            # remove the special chars in review like '"#$@!%^&*()_+-~?>< etc.
            word = ("".join(e for e in words if e.isalnum()))
            # Conver all letters to lower-case
            word = word.lower()
            # stop-word removal
            if not word in stop_words:
                string += word + " "
        data[column][index] = string
```

Fig 6. Preprocessed Apparel Dataset.

```
from sklearn.feature_extraction.text import CountVectorizer
title_vectorizer = CountVectorizer()
title_features   = title_vectorizer.fit_transform(data['title'])
title_features.get_shape()
```

```
(16042, 12609)
```

```
title_features
```

```
<16042x12609 sparse matrix of type '<class 'numpy.int64'>'
        with 147545 stored elements in Compressed Sparse Row format>
```

Fig 7. NLP Preprocessing Algorithm.

```
from sklearn.feature_extraction.text import TfidfVectorizer
tfidf_title_vectorizer = TfidfVectorizer(min_df = 0)
#min_df : When building the vocabulary ignore terms that have a
#document frequency strictly lower than the given threshold.
tfidf_title_features = tfidf_title_vectorizer.fit_transform(data['title'])
tfidf_title_features.get_shape()
```

```
(16042, 12609)
```

```
tfidf_title_features
```

```
<16042x12609 sparse matrix of type '<class 'numpy.float64'>'
        with 147545 stored elements in Compressed Sparse Row format>
```

Fig 8. Dataset BoW Algorithm.

```
idf_title_vectorizer = CountVectorizer()
idf_title_features = idf_title_vectorizer.fit_transform(data['title'])

def n_containing(word):
    return sum(1 for blob in data['title'] if word in blob.split())
def idf(word):
    return math.log(data.shape[0] / (n_containing(word)))

idf_title_features  = idf_title_features.astype(np.float)

for i in idf_title_vectorizer.vocabulary_.keys():
    idf_val = idf(i)
    for j in idf_title_features[:, idf_title_vectorizer.vocabulary_[i]].nonzero()[0]:
        idf_title_features[j,idf_title_vectorizer.vocabulary_[i]] = idf_val
```

Fig 9. Dataset TF-IDF Algorithm.

```
# Set values for various parameters
num_features = 300     # Word vector dimensionality
min_word_count = 1     # Minimum word count
num_workers = 4        # Number of threads to run in parallel
context = 10           # Context window size
downsampling = 1e-3    # Downsample setting for frequent words

# Initialize and train the model (this will take some time)
from gensim.models import word2vec
print ("Training model...")
model = word2vec.Word2Vec(sen_corpus, workers=num_workers, \
            size=num_features, min_count = min_word_count, \
            window = context)
```

Fig 10. Dataset IDF Algorithm.

to find out which can give most information. During analysis, we found out:

- Author columns containsonly one type of value.
- Publisher and manufacture column are almost same as brand attribute.
- Availability column is not useful.
- Reviews don't fetch anything and don't give any information.
- Availability type contains mostly "now" value.

In the dataset Fig. 6, all 7 features have been used for the recommendation. The other features were not used because they were either not useful in recommendation or they don't give information much about product.

- **ASIN**: This feature gives the unique id of amazon product. It is used in fetching the image, and in various NLP functions as index of product.

```
# Set values for various parameters
num_features = 300     # Word vector dimensionality
min_word_count = 1     # Minimum word count
num_workers = 4        # Number of threads to run in parallel
context = 10           # Context window size
downsampling = 1e-3    # Downsample setting for frequent words

# Initialize and train the model (this will take some time)
from gensim.models import word2vec
print ("Training model...")
model = word2vec.Word2Vec(sen_corpus, workers=num_workers, \
            size=num_features, min_count = min_word_count, \
            window = context)
```

Fig 11. Word2Vec Algorithm.

```
doc_id = 0
w2v_title = []
# for every title we build a avg vector representation
for i in data['title']:
    w2v_title.append(build_avg_vec(i, 300, doc_id,'avg'))
    doc_id += 1

# w2v_title = np.array(# number of doc in courpus * 300), each row corresponds to a doc
w2v_title = np.array(w2v_title)
```

Fig 12. Algorithm to create Word2Vec vector.

Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

Fig 13. Example of One Hot Encoding.

```
img_width, img_height = 224, 224

top_model_weights_path = 'bottleneck_fc_model.h5'
train_data_dir = 'images2/'
nb_train_samples = 16042
epochs = 50
batch_size = 1
def save_bottlebeck_features():
    #Function to compute VGG-16 CNN for image feature extraction.
    asins = []
    datagen = ImageDataGenerator(rescale=1. / 255)
    # build the VGG16 network
    model = applications.VGG16(include_top=False, weights='imagenet')
    generator = datagen.flow_from_directory(
        train_data_dir,
        target_size=(img_width, img_height),
        batch_size=batch_size,
        class_mode=None,
        shuffle=False)
    for i in generator.filenames:
        asins.append(i[2:-5])
    bottleneck_features_train = model.predict_generator(generator, nb_train_samples // batch_size)
    bottleneck_features_train = bottleneck_features_train.reshape((16042,25088))
    np.save(open('16k_data_cnn_features.npy', 'wb'), bottleneck_features_train)
    np.save(open('16k_data_cnn_feature_asins.npy', 'wb'), np.array(asins))
save_bottlebeck_features()
```

Fig 14. Algorithm of VGG-16 CNN model.

- **Product_type_name**: This feature is telling the type of product it is. Whether a product is shirt, top, kurta or t-shirt.
- **Formatted price**: This is the price of a product.
- **Color**: This feature tells the color of the product. This will be one-hot encoded and compared to help in recommendation.
- **Brand**: This feature represents the brand of the product. Some users like to view similar type of brands. And the brand vector might help in that.
- **Title**: This feature gives the title of the product. This is the most important feature. All the NLP algorithms will be applied on it. A title of an amazon product contains most of the information, and it usually used in amazon search engines to filter the products.
- **Medium_image_url**: This feature will be used to extract the image from URL given in this feature.

2.2. Data Pre-processing

The brand attribute is the one which is a good attribute and helps a lot in recommendation. It also has very few missing values. And it has many unique values which can be very helpful in providing diverse results.

The color attribute contains missing values. We will be removing those rows where colors are missing. The formatted price and product type also contain some redundancies, we will be removing any such row with these redundancies [23,24].

The title attribute is our main column over which we will use NLP techniques in Fig. 7 to find similarities. Titles of amazon product contains most of the information. Amazon website uses title in product search and encourages sellers to keep most information in those.

On removing rows with missing values, we have reduced our data to 28,000. Reducing data will help us in optimizing quickly.

We will further remove duplicates titles and titles that are short enough to not provide is with information. After removing titles that are very short and titles that are very similar and most likely to have same products, our dataset get reduced to 16,000 products. Finally, our last step would be to remove stop words from our titles. The following is the code used to remove stop words:

2.3. Algorithms and models

The first NLP algorithm that will be used on our dataset is Bag of Words. We will be using a predefined package CountVectorizer( ). It will help us to create a sparse vector of titles of Bag of Words Figs. 8 and 9. We will be using these vectors and compare with all other vectors to find out a similar product.

If the Bag of Words model did not work very well and did not gave good results, thenwe will further use other models and see which model works best. See (Fig. 10 and Fig. 11).

A TF-IDF will be used. TF-IDF will help us finding product that contain special occurrences and put more weightage to less frequent words in titles. A TfidfVectorizer( ) is predefined package that will help us in creating a sparse vector that can be further used to compare and give good results.

TF-IDF Figs. 8 and 9does not always gives correct results. Term frequency might cause problems if a word comes a lot. We want to create something that give importance to words in case they are rare. Hence, we can use an IDF model that will only consider the rarity of a word.

The above algorithm will create a vector that will only create a matrix based of inverse document frequency. The recommendation appeared good but not good enough.

The above algorithms11,12,13 were text-based, we need something that can understand the semantics of text and recommend based on that. We will use word2vec models to do this.

Training our word2vec through our data won't give good results as our data is not big enough. To solve this, we used a google pretrained word2vec. Google trained this with help of google news data. We further fetched a subset from this google word2vec matrix which contains words present only in our titles. This will help in optimizing faster.

The results appear good, but we will use a modified word2vec. Like Bag of Words, we would like to give extra importance to rare

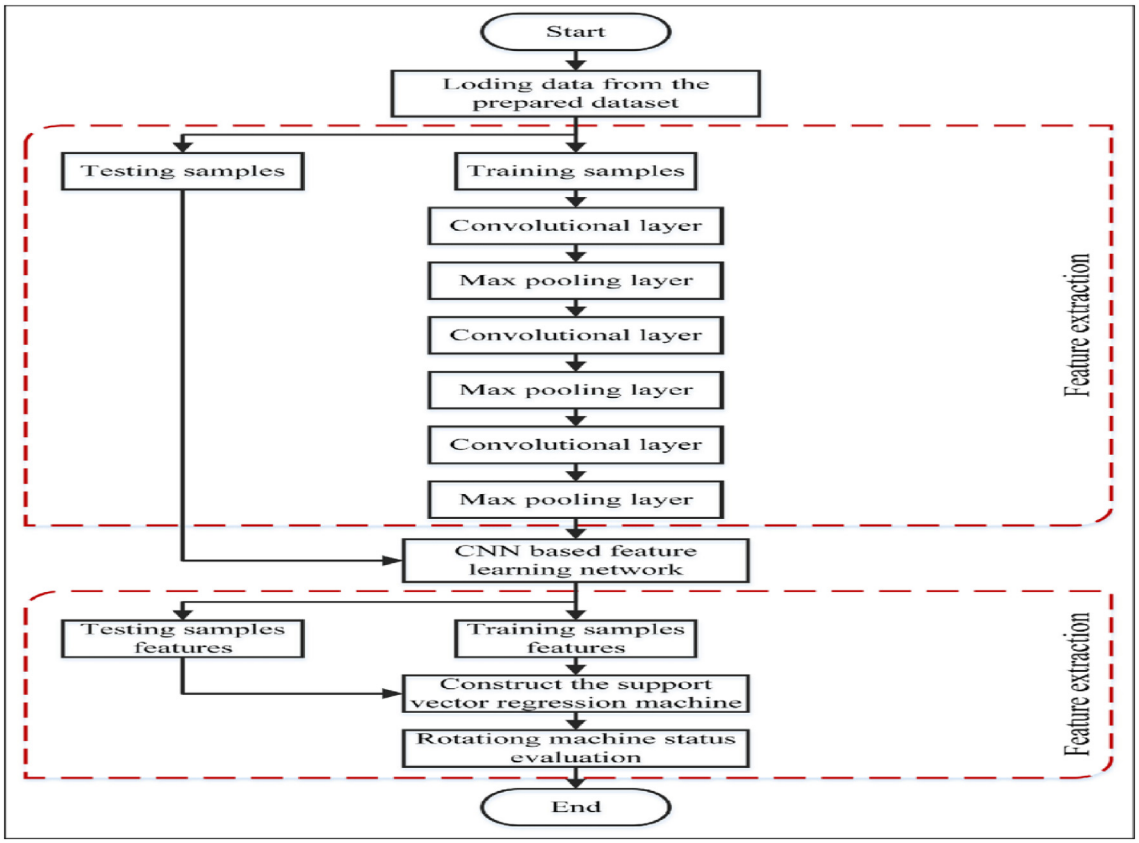


Fig 15. VGG-16 Architecture Flowchart.

words. We will further add IDF weights to each word and then calculate a average. This procedure helps us in getting more clear results.

The IDF-Weighted Word2Vec can be a good model for titles in our dataset.

To make our predictions much better, we will be using other attributes. Brand and Color are categorical attributes, these can be created into a vector using Encoding.

To make our prediction better, we will further create a vector using CNN. We will use pre-trained

VGG-16 model. This model in Figs. 12, 13, 14 will help us to extract features from images and create a vector out of it. VGG-16 has continuous convolution and maxpool layers. These layers are used are feature extraction. The last layer in VGG-16 is classification layer. We will disable the last layer, as we only want the features and not classification.

Finally, as in Figs. 14 and 15 after receiving the vectors of titles, brand, color and images, we will combine these vectors and make one big vector which will represent a single product. These vectors with the help of Euclidean distance will help us in finding the similar products. We will also combine various weights to find out the best combination to get the best results.

3. Implementation

3.1. Modules

The above modules as in Figs. 16 and 17 help in creating a visual presentation of our recommendation images. The function defined are used create vectors and remove any stop words and misspelled words. Also, these function helps in creating the vectors of each product.

The algorithms of TF-IDF model and IDF model represent the data in a form that will help in optimizing and analysis of data that can help us find important features.Various other word2vec models with the algorithm will be helping us in optimizing our data and find out best features.

After the above algorithms, we will use a function that will convert attributes needed to feature vector.Need to create feature vectors by One Hot Encoding

The final algorithm will help us to view the products based on the images. The pairwise distance calculate distance of one product with all the products in the database. We have further used to display the similar products based on images iteratively as shown architecture in Fig. 18.

The VGG-16 CNN architecture is what we used for our feature extraction. VGG has following layers:

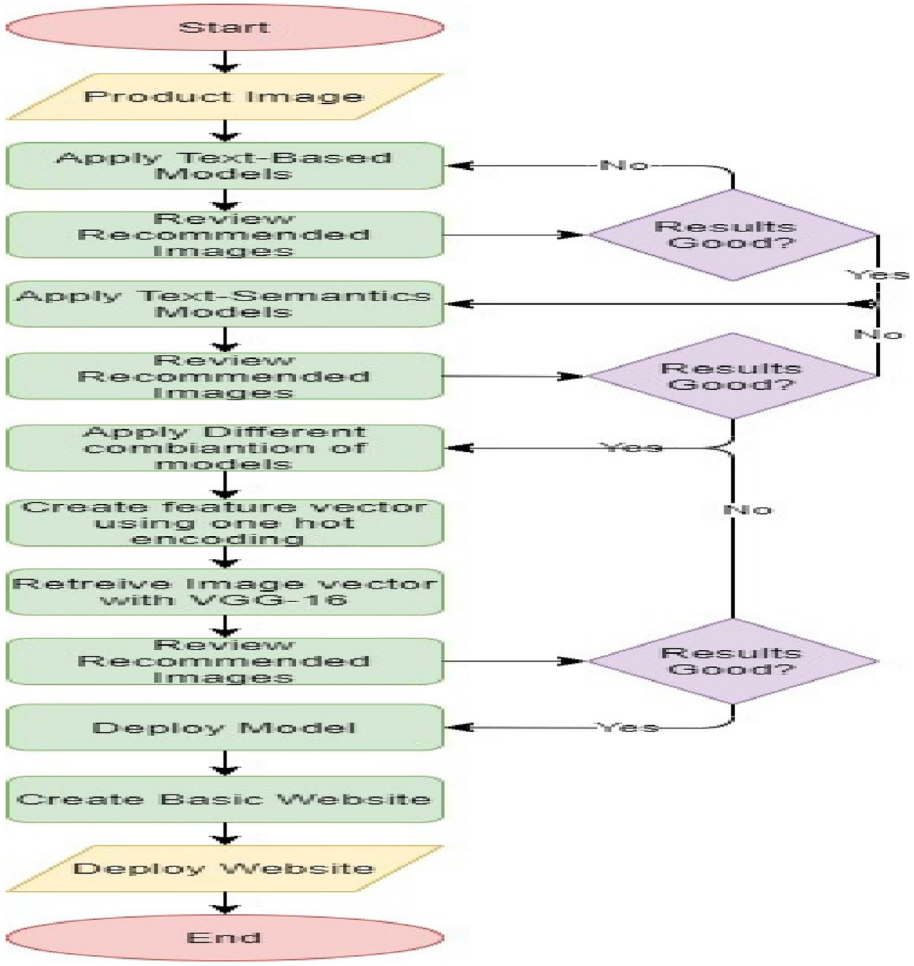


Fig 16. Flow of result to production.

```
def bag_of_words_model(doc_id, num_results):
    pairwise_dist = pairwise_distances(title_features,title_features[doc_id]) #calculate euclidean dist w/ every other
    indices = np.argsort(pairwise_dist.flatten())[0:num_results]
    pdists  = np.sort(pairwise_dist.flatten())[0:num_results]
    df_indices = list(data.index[indices])
    for i in range(0,len(indices)):
        get_result(indices[i],data['title'].loc[df_indices[0]], data['title'].loc[df_indices[i]],//
                   data['medium_image_url'].loc[df_indices[i]], 'bag_of_words')
        print('ASIN :',data['asin'].loc[df_indices[i]])
        print ('Brand:', data['brand'].loc[df_indices[i]])
        print ('Title:', data['title'].loc[df_indices[i]])
        print ('Euclidean similarity with the query image :', pdists[i])
        print('='*60)
```

Fig 17. Algorithm to display BoW results.

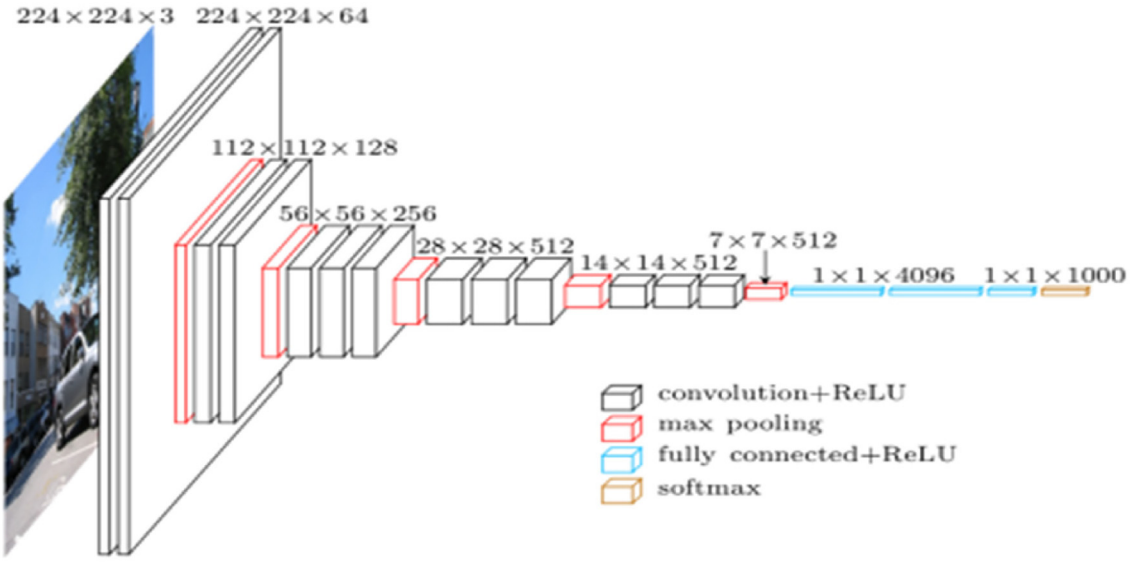


Fig 18. VGG-16 Detailed Architecture [20].

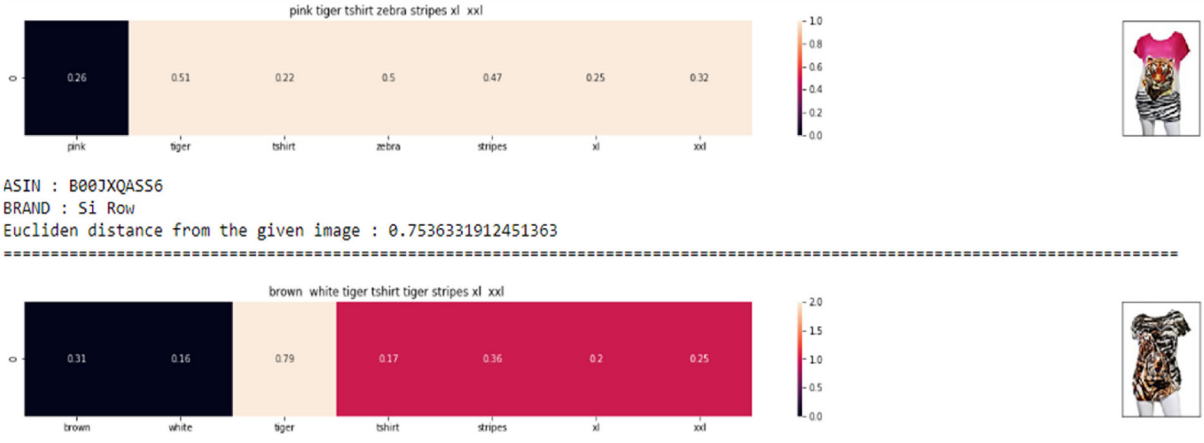


Fig 19. Results of Text Based Recommendation.

2 * convolution based layer of 64 channel of 3x3 kernel.
1 *maxpool based layer of 2x2 pool size.
2 * convolution based layer of 128 channel of 3x3 kernel.
1 * maxpool based layer of 2x2 pool size.
3 * con-volutionbased layer of 256 channel of 3x3 kernel.
1 * maxpool based layer of 2x2 pool size.
3 * con-volutionbased layer of 512 channel of 3x3 kernel.
1 * maxpool based layer of 2x2 pool size.
3 * con-volution layer of 512 channel of 3x3 kernel.
1 * maxpool layer of 2x2 pool size.
A final layer (classification layer) that is used in VGG-16 is:
1 * Dense layer of 4096 units
1 * Dense layer of 4096 units
1 * Dense SoftMax layer of 2 units
We won't be using classification layer in our algorithm.

After finding out the best models, we will create a basic website. This website will just display the products and various features. A user will simply click on the image, and it redirect to second page that will show the similar product. We will use Django to implement our product recommendation website.

We have also used JavaScript, jQuery, CSS, AJAX to implement [11,19]website with better functionalities and make it more presentable.

3.2. Prototypes

3.2.1. Text based recommendations

The algorithms we created, are used to display the results in this formation as in Fig. 19. The above represents a heatmap of the words present in title. The value in each word represent TF-IDF value of each word.

The color coded in on each word represent how much a word effects the query title. While a dark shade tells word is completely opposite, a lighter shade tells that word is present in query title as well. This heatmap will further help us understand why a product is more likely to be selected as recommendation.

3.2.2. Text semantics recommendations

The text semantic algorithms will represent recommended models and we have used a heatmap to find out better results. The heatmap tells how words in query image and recommended image, are semantically related. This way we can understand the results in Fig. 20 of recommender system.

The above Figs. 21 and 22 is the basic prototype of working website for our recommendation system. The website gives basic information the products available. On clicking, it redirects the user to another webpage which displays the products that are like the selected product.

4. Results

Our product recommendation system can predict products that were very similar to selected products. The model and algorithm selected was giving good results. We will need to do AB testing to find out better results, which can be done with enough users.

Through manual analysis, the product appears to be from similar brands, similar types and similar colors which tells models is working well. Also, the image similarity further helps in getting good results.

In the fig, the first product is query image, and subsequent results are the product recommended by our system. The recommender system shows result that are from same brands and are very similar to query product in terms of product type and similar pattern.

Our product recommendation Figs. 23, 24, 25 system will be deployed as a website. This website will work to recommend user the products that are like it. It is a basic website working Django that contain algorithms that will display the results and all the various attributes related to it.

5. Conclusions

The Product Recommendation System we developed is still in progress and has much scope of improvement. It has a good result while it has not taken YES/NO type questions into account. It is very effective for content-based recommendations.

Fig 20. Results of Text Semantics Recommendation.

Apparel Recommendation

Select A Product

Image	Name	Brand	Color
	active basic womens basic casual plain camisole cami top tank teal medium	Active Products	Teal
	alternative womens meegs printed racerback ecojersey tank top small stars	Alternative	Stars
	annakaci sm fit brown three fashion friends graphic print paisley panel top	Anna-Kaci	Brown
			

Results are

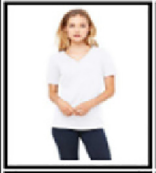
Image	Name	Brand
	fifteen twenty womens shoulder button top sz small	FIFTEEN TWENTY
	active basic womens basic casual plain camisole cami top tank teal medium	Active Products
	bella canvas missys relaxed jersey shortsleeve vneck tshirt 6405 whitexlarge	Bella
		

Fig 21. Prototype Website – Display Products.

ASIN: B00JXQB5FQ
Product Title: burnt umber tiger tshirt zebra stripes xl xxl
Brand : Si Row
Euclidean Distance from input image: 1.4024722938088781
Amazon Url: www.amzon.com/dp/B00JXQB5FQ

ASIN: B00JXQASS6
Product Title: pink tiger tshirt zebra stripes xl xxl
Brand : Si Row
Euclidean Distance from input image: 14.981206655469132
Amazon Url: www.amzon.com/dp/B00JXQASS6

ASIN: B00JXQCUIC
Product Title: yellow tiger tshirt tiger stripes l
Brand : Si Row
Euclidean Distance from input image: 19.658605091519192
Amazon Url: www.amzon.com/dp/B00JXQCUIC

Fig 22. Prototype Website – Display Recommendations.

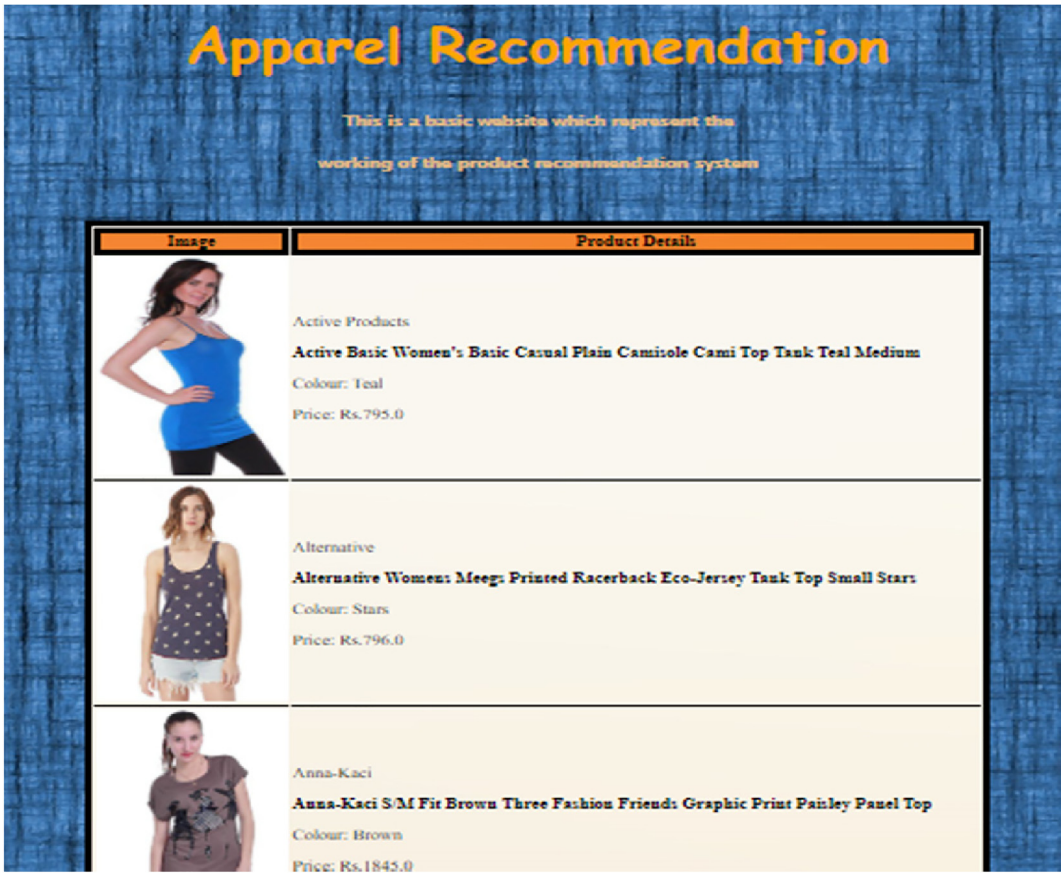


Fig 23. Final Model Results.

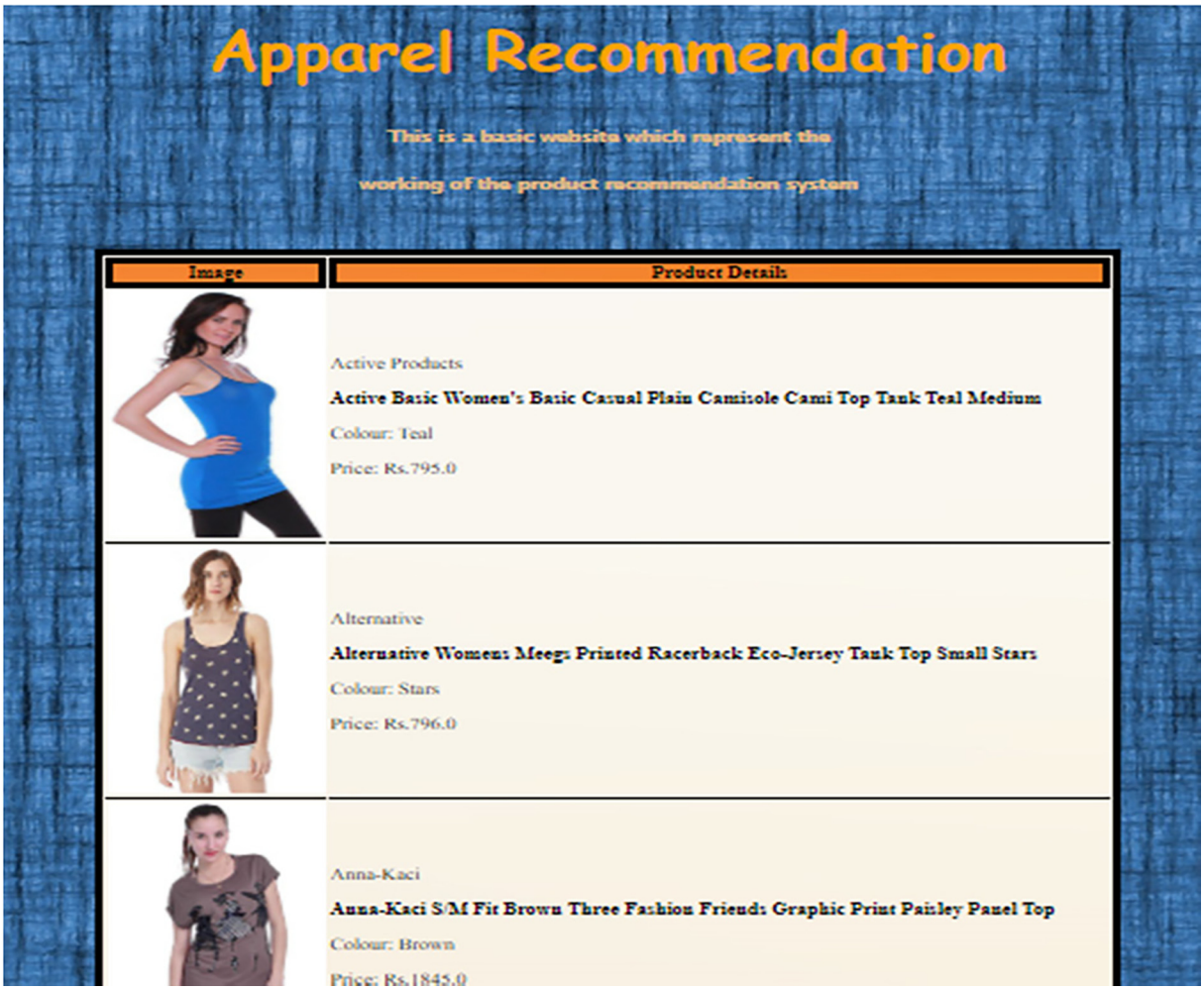


Fig 24. Website – Products.

Thus, we can conclude that the algorithms we user are very efficient and good at NLP tasks and recommending, embedding, and other functions significantly reduced our code and made our Recommendation System faster.

We have also seen how our model can be used for end-users and developed a beginning of a website portal for it, that users can select their product. Though at present it is to only limited products.

6. Future scope

The proposed model has shown it can be used as an effective Product Recommendation System, but it still has a lot it can improve upon. We can improve testing procedure. We have done testing manually. We will need to AB testing to get better results. We can in future work on the website. We can implement various features like product search. We can collaborate our model with collaborative filtering to get better results. For this we will need different users or user data. We can also use different type of attributes or use a method to fill up the null values based on machine learning model. We can also use description, reviews and ratings of a product to get better results.

CRediT authorship contribution statement

Akhilesh Kumar Sharma: Conceptualization, Data curation. **Bhavna Bajpai:** Formal analysis, Funding acquisition. **Rachit Adhvaryu:** Investigation, Methodology, Project administration. **Suthar Dhruvi Pankajkumar:** Resources, Software, Supervision. **Prajapati Parthkumar Gordhanbhai:** Validation, Visualization. **Atul Kumar:** Writing - original draft, Writing - review & editing.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Fig 25. Website - Recommendations.

REFERENCES

[1] Introduction to Word2Vec, TowardsDataScience.
[2] A Gentle Introduction to the Bag-of-Words Model, MachineLearningMastery.
[3] TF-IDF from scratch in python on real world dataset, TowardsDataScience.
[4] Step by step VGG16 implementation in Keras for beginners, TowardsDataScience.
[5] D. Das, L. Sahoo, S. Datta, A survey on recommendation system, Int. J. Computer Appli. 160 (7) (2017) 6–10.
[6] M.-C. Chiu, J.-H. Huang, S. Gupta, G. Akman, Developing a personalized recommendation system in a smart product service system based on unsupervised learning model, Computers Indus. 128 (2021) 103421, https://doi.org/10.1016/j.compind.2021.103421.
[7] M. Aamir, M. Bhusry, Recommendation system: state of the art approach, Int. J. Computer Appl. 120 (12) (2015) 25–32.
[8] Y. Zhang et al., Towards conversational search and recommendation: system ask, user respond, Proceedings of the 27th acm international conference on information and knowledge management, 2018.
[9] S. Akram et al., ChoseAmobile: a web-based recommendation system for mobile phone products, J. Internet Technol. 21 (4) (2020) 1003–1011.
[10] R. Albalawi, T.H. Yeap, M. Benyoucef, Toward a real-time social recommendation system, Proceedings of the 11th International Conference on Management of Digital EcoSystems. 2019.
[11] R.V. Karthik, S. Ganapathy, A fuzzy recommendation system for predicting the customers interests using sentiment analysis and ontology in e-commerce, Appl. Soft Computing 108 (2021) 107396, https://doi.org/10.1016/j.asoc.2021.107396.
[12] A.R. Sulthana, S. Ramasamy, Ontology and context based recommendation system using neuro-fuzzy classification, Computers Electrical Eng. 74 (2019) 498–510.
[13] S. Sharda, G.S. Josan, Machine learning based recommendation system: a review, Int. J. Next-Gener. Comput. 12 (2021) 2.
[14] L. Karthikeyan, T. Vetriselvi, Journal Recommendation System Using NLP, CNN and RNN.
[15] V. Subramaniyaswamy, et al., Data mining-based tag recommendation system: an overview, Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery 5.3 (2015): 87-112.
[16] A. Da'u, N. Salim, Recommendation system based on deep learning methods: a systematic review and new directions, Arti. Intell. Rev. 53 (4) (2020) 2709–2748.
[17] A. Da'u, N. Salim, R. Idris, An adaptive deep learning method for item recommendation system, Know.-Based Syst. 213 (2021) 106681, https://doi.org/10.1016/j.knosys.2020.106681.
[18] S. Gayathri, K. Thyagarajan, Sentiment analysis for product recommendation system using enhanced stochastic learning algorithm, 2019.
[19] V.P. Muhsina, C. Naseer, Domain specific publication recommendation system using deep learning approach.
[20] M. Kalpana, A Deep Learning Sentiment Based Intelligent Product Recommendation System, 2019.
[21] N. Gunjal Sanjay, Development of recommendation system using hybrid approach, 2020.
[22] K. Simonyan, A. Zisserman, Very deep convolutional networks for large-scale image recognition, arXiv.org, April 2015.
[23] A.K. Sharma, et al., Classification of indian classical music with time-series matching using deep learning, in IEEE Access, doi: 10.1109/ACCESS.2021.3093911.
[24] P. Ramani, N. Pradhan, A.K. Sharma, Classification algorithms to predict heart diseases—a survey, Springer, Singapore, 2020.
[25] T. Limbasiya, M. Soni, S.K. Mishra, Advanced formal authentication protocol using smart cards for network applicants, Computers & Electrical Engineering, Volume 66,2018,Pages 50-63,ISSN 0045-7906.
[26] M. Soni, D. Kumar, Wavelet based digital watermarking scheme for medical images, 2020 12th International Conference on Computational Intelligence and Communication Networks (CICN), Bhimtal, India, 2020, pp. 403-407, doi: 10.1109/CICN49253.2020.9242626.
[27] M. Soni, D.K. Singh, Privacy preserving authentication and key management protocol for health information system, Data Protection and Privacy in Healthcare: Research and Innovations, Page-37, CRC Publication,2021.

[28] M. Soni, D.K. Singh, Blockchain-based security & privacy for biomedical and healthcare information exchange systems,Materials Today: Proceedings,2021, ISSN 2214-7853,https://doi.org/10.1016/j.matpr.2021.02.094.
[29] M. Soni, D.K. Singh, LAKA: lightweight authentication and key agreement protocol for internet of things based wireless body area network, Wireless Pers Commun. (2021), https://doi.org/10.1007/s11277-021-08565-2.
[30] M. Soni, Y. Barot, S. Gomathi, A review on Privacy-Preserving Data Preprocessing, Journal of Cybersecurity and Information Management, Volume 4, Issue 2, Page 16-30.
[31] M. Soni , T. Patel, A. Jain, Security analysis on remote user authentication methods. In: Pandian A., Senjyu T., Islam S., Wang H. (eds) Proceeding of the International Conference on Computer Networks, Big Data and IoT (ICCBI - 2018). ICCBI 2018. Lecture Notes on Data Engineering and Communications Technologies, 2020, vol 31. Springer, Cham. https://doi.org/10.1007/978-3-030-24643-3_60.
[32] M. Patel , D. Rami, M. Soni, Next generation web for alumni web portal, In: Balaji S., Rocha Á., Chung YN. (eds) Intelligent Communication Technologies and Virtual Mobile Networks. ICICV 2019. Lecture Notes on Data Engineering and Communications Technologies, 2020, vol 33. Springer, Cham. https://doi.org/10.1007/978-3-030-28364-3_16.
[33] M. Soni, A. Jain, Secure communication and implementation technique for sybil attack in vehicular Ad-Hoc networks, 2018 Second International Conference on Computing Methodologies and Communication (ICCMC), Erode, 2018, pp. 539-543, doi: 10.1109/ICCMC.2018.8487887.
[34] M. Soni, B.S. Rajput, T. Patel, N. Parmar, Lightweight vehicle-to-infrastructure message verification method for VANET. In: Kotecha K., Piuri V., Shah H., Patel R. (eds) Data Science and Intelligent Applications. Lecture Notes on Data Engineering and Communications Technologies, 2021, vol 52. Springer, Singapore. https://doi.org/10.1007/978-981-15-4474-3_50.
[35] U. Chaudhary, A. Patel, A. Patel, M. Soni, Survey paper on automatic vehicle accident detection and rescue system. In: Kotecha K., Piuri V., Shah H., Patel R. (eds) Data Science and Intelligent Applications. Lecture Notes on Data Engineering and Communications Technologies, 2021, vol 52. Springer, Singapore. https://doi.org/10.1007/978-981-15-4474-3_35.
[36] M. Soni, B.S. Rajput, Security and performance evaluations of QUIC protocol. In: Kotecha K., Piuri V., Shah H., Patel R. (eds) Data Science and Intelligent Applications. Lecture Notes on Data Engineering and Communications Technologies, 2021, vol 52. Springer, Singapore. https://doi.org/10.1007/978-981-15-4474-3_51.
[37] M. Soni, A. Jain, T. Patel, Human movement identification using Wi-Fi signals, 2018 3rd International Conference on Inventive Computation Technologies (ICICT), Coimbatore, India, 2018, pp. 422-427, doi: 10.1109/ICICT43934.2018.9034451.

Further Reading

[1] T. Michelle, Machine Learning Marching Learning, McGraw Hill Education; First edition.
[2] C. SehrishMunawar, et al., A recommendation system for functional features to aid requirements reuse, 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET). IEEE, 2020.
[3] B. Vaishali, Y. Yadav, An improved E-commerce product recommendation system using weighted technique.
[4] V. Bajpai, Y. Yadav, An improved dynamic e-commerce recommendation system, Proceedings of Recent Advances in Interdisciplinary Trends in Engineering & Applications (RAITEA) (2019).