RECOMMENDATION STSTEM FOR ONLINE PRODUCTS

[1]M.Tech Student, Department of Computer Science Engineering, Jogaiah Institutes of Technology and Sciences, National Highway 214, Kalagampudi, Dist, Palakollu, Andhra Pradesh 534268
[2] Assistant Professor, Department of Computer Science Engineering, Jogaiah Institutes of Technology and Sciences, National Highway 214, Kalagampudi, Dist, Palakollu, Andhra Pradesh 534268
[1]BANDARU SATEESH, [2]K.TULASI RAM

ABSTRACT

On the Internet, where the number of choices is overwhelming, there is a need to filter, prioritize and efficiently deliver relevant information in order to alleviate the problem of information overload, which has created a potential problem for many Internet users. Recommender systems solve this problem by searching through large volumes of dynamically generated information to provide users with personalized content and services. In view of more personalized clothing requirements, an intelligent clothing recommendation system wasdesigned and developed in this project. By the use of Transfer Learning to elicit the rich information from the product images, and the use of cosine similarity approach, the user is provided with eclectic recommended products depending on their choices. This was implemented on a web e-commerce application, where users havea wide range to choose their clothing apparel from our database. Only those images are recommended to users which have 80% or above similarity with the product chosen by the user, which helps in solving personalized recommendations.

1.1 INTRODUCTION

FLIPKART use recommender systems to suggest products to their customers. Meanwhile, social networks such as LinkedIn and FACEBOOK use them to suggest new contacts. It takes exclusively into account the expressed opinion about the other items in order to make recommendations. Meanwhile, content-based filtering uses the knowledge it has of the items and their attributes to make recommendations.

1.2 HISTORY

The systems originally started in the 1990s based on the widespread research progress in Collective Intelligence. During this period, recommendations were generally provided to consumers based on their rating structure. The first consumer-focused recommendation system was developed and commercialized by Goldberg,Nichols, Oki and Terry in 1992. Tapestry, an electronic messaging system was developed to allow users only to rate messages as either a good or bad product and service.

However, now there are plenty of methods to obtain information about the consumer's liking for a product through the Internet. These data can be retrieved in the forms of voting, tagging, reviewing and the number of likes or dislikes the user provides. It may also include reviews written in blogs, videos uploaded on YouTube or messages about a product. Regardless of communication and presentation, medium preferences are expressed in the form of numerical values.

1.3 SCOPE

With the advent of emerging technologies and the rapid growth of the Internet, the world is moving towards an E world where most things are digitized and available with a mouse click. E-Commerce is steadily becoming more important in changing the way people buy products and services. Customers are buying more andmore products on the website. Whenever a customer wants to buy a product on the web, they visit an online storeand look for items of their interest.

1.4 PROBLEM STATEMENT

Fashion Recommendation System can be defined as a means of feature matching between fashion products and users or consumers under specific matching criteria. Different research addressed apparel attributes such as the formulation of colors, clothing shapes, outfit or styles, patterns or prints and fabric structures or textures. In our system we need to develop a recommendation system that predicts recommendations based on color compatibility and feature extraction.

1.5 EXISTING SYSTEM:

Most existing recommendation systems focus on fashion and clothing only. In this paper, we propose an online recommendation system based on color compatibility for textile products. The system recommends a compatible pattern to consumers on a new product to be purchased with the existing products. Whether the patterns are compatible or not was evaluated according to the feedback received from the designers and participants. Unlike traditional recommendation system methods, CNNs were used in the study. To our best knowledge, there is no study that considers pattern recommendation considering color compatibility in textile products using deep learning algorithms. We think that our proposed study is the first in the literature.

1.6 PROPOSED SYSTEM:

Most existing recommendation systems focus on fashion and clothing with text based approach the proposed system has an online recommendation system based on color compatibility for textile products. The system recommends compatible patterns to consumers on a new product to be purchased with the existing products. Whether the patterns are compatible or not was evaluated according to the feedback received from thedesigners and participants. Unlike traditional recommendation system methods, CNNs were used in the study. There is no study that considers pattern recommendation considering color compatibility in textile products using deep learning algorithms.

EXISTING SYSTEM	PROPOSED SYSTEM
1. IMAGENET module is used.	1. RESNET module is used.
2. It is not a Pertained module.	2. It is a Pretrained module.
3. Less dataset used.	3. Huge dataset is used (44k images).
4. Extracts less features.	4. It extracts 2048 features for every image.
5. Low accuracy.	5. Accuracy up to 95%.

Table 1.1: Comparison Values between Existing System and Proposed System

1.7 ADVANTAGES:

- Developed recommendation system was highly responsive and speed in performance.
- More number of data files are added and is more accurate while producing outputs.
- It has the capability to include color compatibility feature for the recommended products.

2. LITERATURE SURVEY

- **A Review of Modern Fashion Recommender Systems** by *YASHAR DELDJOO, FATEMEH NAZARY, ARNAU RAMISA, JULIAN MCAULEY, GIOVANNI PELLEGRINI, ALEJANDRO BELLOGIN, TOMMASO DI NOIA.*

- **Building a Recommendation System Using Neural Network Embeddings by** *WILL KOERSHEN, TOM MITCHELL, CLEOMAR VALOIS B. JR, MARCIUS ARMADA DE OLIVEIRA*.

- **Deep Learning based Recommender System: A Survey and New Perspectives by** SHUAI ZHANG, LINO YAVO, AIXIN SUN, YI TAY

3.1 REQUIREMENT ANALYSIS

The process to gather the software requirements from

clients, analyze and document them is known as requirements engineering or requirements analysis. The goal of requirement engineering is to develop and maintain sophisticated and descriptive 'System/Software Requirements Specification' documents. It is a four-step process generally, which includes –

- Feasibility Study four-step.
- Requirements Gathering.
- Software Requirements Specification.
- Software Requirements Validation.

3.2 PROCESS MODE USED:

The model that is basically being followed is the WATERFALL MODEL, which states that the phases are organized in a linear order. First of all the feasibility study is done. Once that part is over the requirement analysis and project planning begins. If a system exists and modification and the addition of a new module are needed, analysis of the present system can be used as a basic model. The design starts after the requirement analysis is complete and the coding begins after the design is complete. Once the programming is completed, the testing is done. In this model, the sequence of activities performed in a software development project is Requirement Analysis, Project Planning, System Design, Detail design, Coding, Unit testing, System integration & testing. Here the linear ordering of these activities is critical. The end of the phase and the output of one phaseis the input of the other phase.

The output of each phase is to be consistent with the overall requirement of the system. Some of the qualities of the spiral model are also incorporated after the people concerned with the project review completion of each of the phases of the work done. WATERFALL MODEL was chosen because all requirements were known beforehand and the objective of our software development is the computerization/automation of an already existing manual working system.

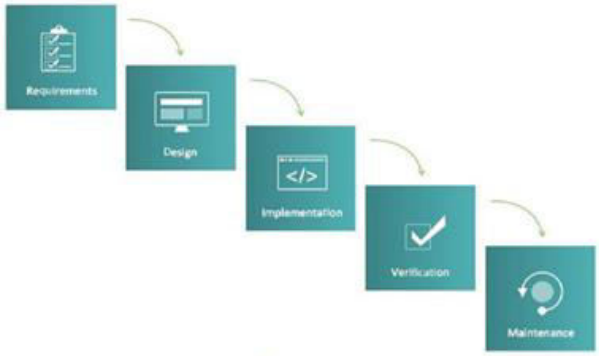

Fig 1: Process model

4.1 Software Requirements Specification (SRS):

A requirements specification for a software system- is a complete description of the behavior of a system to be developed. It includes a set of cases that describe all the interactions users will have with the software. In addition to use cases, the SRS also contains non-functional requirements. Non-functional requirements are requirements that impose constraints on the design or implementation (such as performance engineering requirements, quality standards, or design constraints).

System Requirements Specification It is a collection of information that embodies the requirements of a system. A business analyst, sometimes titled system analyst, is responsible for analyzing the business needs of their clients and stakeholders to help identify business problems and propose solutions. Projects are subject to three sorts of required elements. Business requirements describe in business terms what must be delivered or accomplished to provide value.

- Product requirements describe properties of a system or product (which could be one of several ways to accomplish a set of business requirements.)
- Process requirements describe activities performed by the developing organization. For instance, process requirements could specify methodologies that must be

followed, and constraints that the organization must obey.

Product and process requirements are closely linked. Process requirements often specify the activities that will be performed to satisfy a product requirement. For example, a maximum development cost requirement (a process requirement) may be imposed to help achieve a maximum sales price requirement (a product requirement)a requirement that the product is maintainable (a product requirement) often is addressed by imposing requirements to follow particular development styles. A system engineering requirement can be a description of what a system must do, referred to as a Functional Requirement. This type of requirement specifies something that the delivered system must be able to do. Another type of requirement specifies something about the system itself, and how well it performs its functions. Such requirements are often called Nonfunctional requirements, or 'Performance requirements' or 'Quality of service requirements'. Examples of such requirements include usability, availability, reliability, supportability, testability, and maintainability.

A collection of requirements define the characteristics or features of the desired system. A 'good' list of requirements as far as possible avoids saying how the system should implement the requirements, leaving such decisions to the system designer. Specifying how the system should be implemented is called "implementation bias" or "solution engineering". However, implementation constraints on the solution may validly be expressed by the future owner, for example for required interfaces to external systems; for interoperability with other systems; and for commonality with other owned products.

4.2 Functional requirements:

The Functional Requirements Specification gives the operations and activities that a system must be able to perform. Functional requirements should include functions performed by specific screens, outlines of workflows performed by the system, and other business or compliance requirements the system must meet. It also depends upon the type of software, expected users, and the type of system where the software is used. Some constraints were given as input.

4.3 Non-functional requirements:

In systems engineering, a non-functional requirement is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. The nonfunctional requirements can be considered as quality attributes of a system .NFRs are persistent qualities and constraints that, unlike functional requirements, are typically revisited as part of the Definition of Done (DoD) for each Iteration, Program Increment (PI), or release. NFRs influence all backlogs: Team, Program, Solution, and Portfolio.

- **Performance**: The time required to predict the accuracy.
- **Reliability:** The system should be 90% reliable. Since it may need some maintenance or preparation for someparticular day, the system does not need to be reliable every time. so, 80% reliability is enough.
- **Efficiency**: Based upon the density of given values or input to predict.
- **Availability**: It is available and getting updated day by day.
- **Maintainability**: The system should be optimized for supportability, or ease of maintenance as far as possible.

5. DESIGN PHASE

Design is a multi-step process that focuses on data structure, Software architecture, procedural details, and interface between modules. The design process also

translates the requirements into the presentation of software that can be accessed for quality before coding begins. Computer software design changes continuously as new methods; better analysis and broader understanding evolved. Software design is at a relatively early stage in its revolution. Therefore, software design methodology lacks the depth, flexibility, and quantitative nature that are normally associated with more classical engineering disciplines. However, the techniques for software design do exist, criteria for design qualities are available and design notation can be applied. The purpose of the design phase is to plan a solution to the problem specified by the requirements document. The design of a system is perhaps the most critical factor affecting the quality of the software. It has amajor impact on the project during later phases, particularly during testing and maintenance.

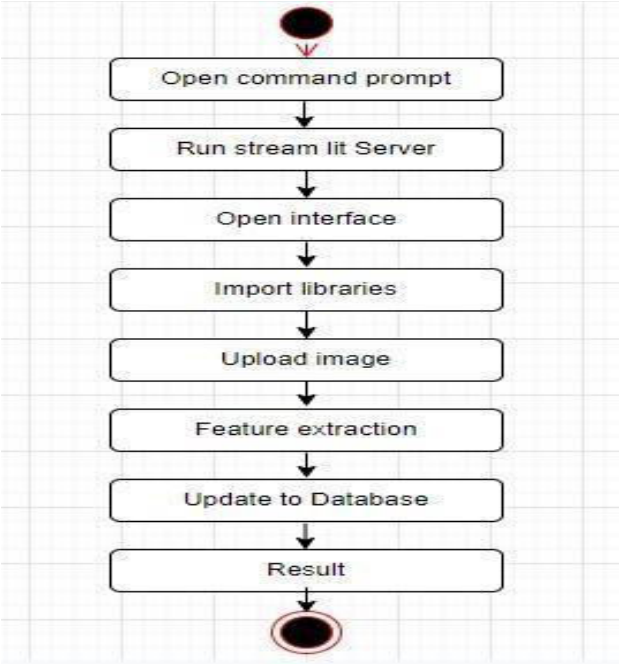

Fig 2: Flow Chart for Execution

6.1 SYSTEM DESIGN

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent asystem in terms of input data to the system, various processing carried out on this data, and the output datais generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the systemcomponents. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

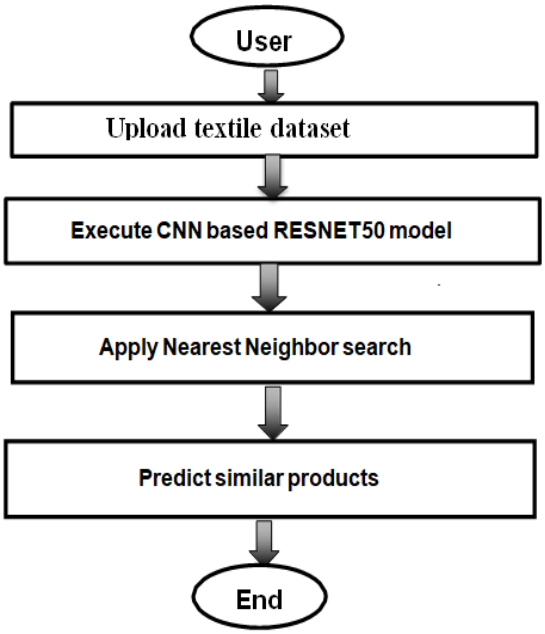

Fig 3: System Design

6.2 USE CASE DIAGRAM FOR RECOMMENDATION SYSTEM

UML is an acronym that stands for Unified Modeling Language. Simply put, UML is a modern approach to modeling and documenting software. In fact, it's one of the most popular business process modeling techniques.

It is based on diagrammatic representations of software components. As the old proverb says: "a picture is worth a thousand words". By using visual representations, we are able to better understand possible flaws or errors in software or business processes. There were several different ways to represent and document software systems. The need arose for a more unified way to visually represent those systems and as a result, the UML was developed by three software engineers working at Rational Software. It was later adopted as the standard and has remained the standard ever since, receiving only a few updates.

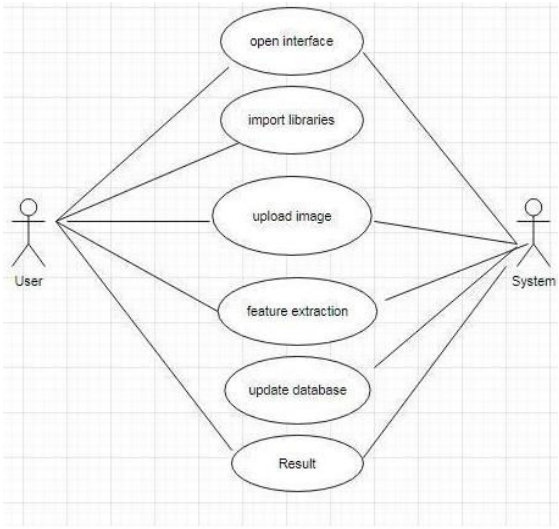

Fig 4: Use Case Diagram for Online Recommendation System

6.3 CLASS DIAGRAM FOR RECOMMENDATION SYSTEM:

Class diagrams give an overview of a system by showing its classes and the relationships among them. Class diagrams are static – they display what interacts but not what happens when they do interact. In general a class diagram consists of some set of attributes and operations. Operations will be performed on the data values of attributes.

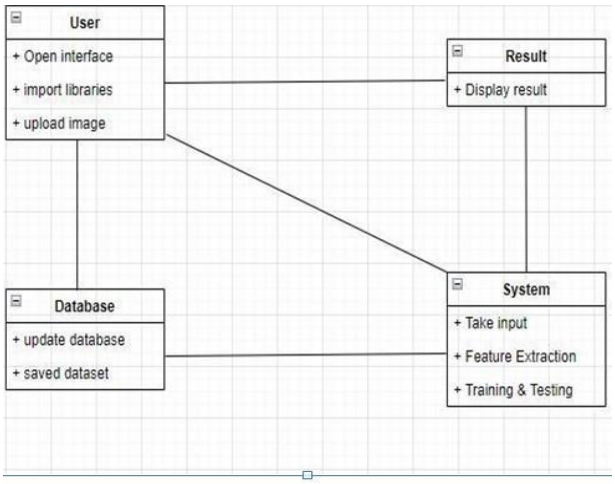

Fig 5: Class Diagram for Online Recommendation System

Algorithmic Design:

Step 1: Start

Step 2: Open interface Step

3: Upload data setsStep 4:

Pre-processing

Step 5: Training and testingStep

6: Enter the constraintsStep 7:

Submit

Step 8: Get the resultStep

9: Stop

6.4SEQUENCE DIAGRAM FOR RECOMMENDATION SYSTEM:

UML Sequence Diagrams are designed so that they can depict a timeline. On the top, you can see the beginning, and then the diagram flow descends downwards to mark the sequence of all interactions in the system.These interactions and objects have some symbols and notations that are used to standardize UML Structure Diagram.

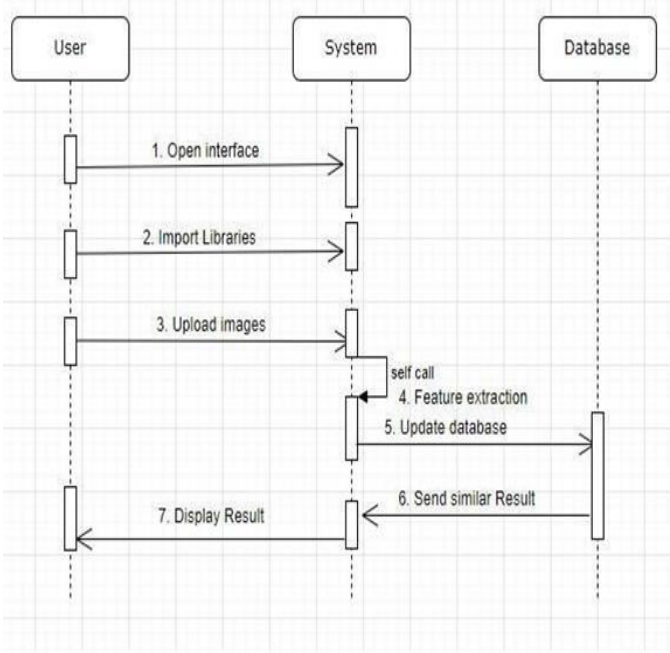

Fig 6: Sequence Diagram for Recommendation System

7.1 IMPLEMENTATION:

1st the user need to load the data sets for the training After loading the data sets the system will preprocessthe data set images and build the model. After building the model the system will print the data set and test them and find the accuracy of the model and at last it will save the model. The user need to open the source folder, Run the python file with the name " app.py". The assistant will redirect to the Web interface.

IMPLEMENTATION STEPS:

Step 1: Start

Step 2: Open interface Step

3: Upload data setsStep 4:

Pre-processing

Step 5: Training and testingStep

6: result

Step 7: stop

CODING (SOURCE CODE)

```
import streamlit as st
import tensorflow import
pandas as pd from PIL
import Imageimport pickle
import numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.resnet50 import ResNet50,
Preprocess inputfrom
tensorflow.keras.layers import GlobalMaxPooling2D
from tensorflow.keras.models import Sequentialfrom
numpy.linalg import norm
from sklearn.neighbors import NearestNeighborsimport os
features_list = pickle.load(open("image_features_embedding.pkl", "rb"))
img_files_list = pickle.load(open("img_files.pkl", "rb"))
model = ResNet50(weights="imagenet", include_top=False,
 input_shape=(224, 224, 3))model.trainable = False
model = Sequential([model, GlobalMaxPooling2D()])
st.title('Recommendation system for online products')
def save_file(uploaded_file):try:
with open(os.path.join("uploader", uploaded_file.name),'wb') as
f:f.write(uploaded_file.getbuffer())
            return 1
    except:
        return 0
```

```
display image
show_images = Image.open(uploaded_file)
size = (400, 400)
resized_im = show_images.resize(size)
st.image(resized_im)
# extract features of uploaded image
features = extract_img_features(os.path.join("uploader",  uploaded_file.name),
model)#st.text(features)
img_indicess = recommendd(features,  features_list)
col1,col2,col3,col4,col5  = st.columns(5)
with col1:st.header("I")
st.image(img_files_list[img_indicess[0][0]])
with col2:
    st.header("II")
    st.image(img_files_list[img_indicess[0][1]])
with col3:
    st.header("III")
    st.image(img_files_list[img_indicess[0][2]])
with col4:
    st.header("IV")
    st.image(img_files_list[img_indicess[0][3]])
with col5:
    st.header("V")
    st.image(img_files_list[img_indicess[0][4]])
    else:
st.header("Some error occur")
```

```
def extract_img_features(img_path, model):
img = image.load_img(img_path, target_size=(224, 224))
img_array = image.img_to_array(img)
expand_img = np.expand_dims(img_array, axis=0)
preprocessed_img = preprocess_input(expand_img)
result_to_resnet = model.predict(preprocessed_img)
flatten_result = result_to_resnet.flatten()
# normalizing
result_normlized = flatten_result / norm(flatten_result)
return result_normlized
def recommendd(features, features_list):
neighbors = NearestNeighbors(n_neighbors=6,
algorithm='brute', metric='euclidean')neighbors.fit(features_list)
distence, indices = neighbors.kneighbors([features])return indices
uploaded_file = st.file_uploader("Choose your image")
if uploaded_file is not None:
if save_file(uploaded_file):#
```

```
import pickle import
numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.resnet50 import ResNet50,
preprocess_inputfrom tensorflow.keras.layers import GlobalMaxPooling2D
from tensorflow.keras.models import Sequentialfrom
numpy.linalg import norm
from sklearn.neighbors import NearestNeighborsimport
cv2features_list =
pickle.load(open("image_features_embedding.pkl", "rb"))
img_files_list = pickle.load(open("img_files.pkl",
"rb"))print(np.array(features_list).shape)model =
ResNet50(weights="imagenet", include_top=False,
input_shape=(224, 224, 3))model.trainable = False model
= Sequential([model, GlobalMaxPooling2D()])img =
image.load_img('sample/shoes.jpg',target_size=(224,224))
img_array = image.img_to_array(img)
expand_img = np.expand_dims(img_array,axis=0)
```

```
preprocessed_img = preprocess_input(expand_img)
result_to_resnet = model.predict(preprocessed_img)
flatten_result = result_to_resnet.flatten()
# normalizing
result_normlized = flatten_result / norm(flatten_result)
neighbors = NearestNeighbors(n_neighbors = 6, algorithm='brute',
metric='euclidean')neighbors.fit(features_list)
distence, indices = neighbors.kneighbors([result_normlized])print(indices)
for file in indices[0][1:6]:
    print(img_files_list[file])
    tmp_img = cv2.imread(img_files_list[file])
    tmp_img = cv2.resize(tmp_img,(200,200))
    cv2.imshow("output", tmp_img)
    cv2.waitKey(0)
```

PACKAGES IMPORTED

Package:

1. Streamlet
2. Numpy
3. Tensorflow
4. Pandas
5. Pickle
6. PIL

7.1 INTRODUCTION TO TESTING:

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies, and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. Thereare various types of tests. Each test type addresses a specific testing requirement.

7.2 TYPES OF TESTING:

- **White Box Testing:**

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure, and language of the software, or at least its purpose. It is purposeful. It is used to test areas that cannot be reached from a black box level.

- **Black Box Testing:**

Black Box Testing is testing the software without any knowledge of the inner workings, structure, or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a test in which the software under test is treated as a black box. You cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.

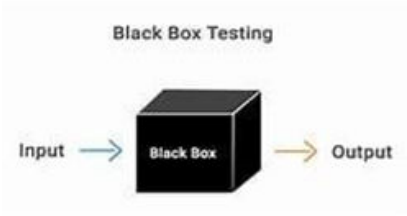

Fig 7: Black Box Testing

S.NO	INPUT	OUTPUT	RESULT
Test Case 1	The user gives the input in the form of training with a dataset.	An output is predicted asthe training is successful	The result is that the dataset is trained. Therefore the test case1 is passed successfully.
Test Case 2	The user gives the inputin the form of Opening interface.	An output is predicted as the user opens the interface is successful.	The result is that the user opened the interface. Therefore thetestcase2 passed Successfully.
Test Case 3	The user gives the input in the form of uploading an image.	An output is predicted as the user uploads the imageis successful.	The result is that the user uploads the image. Therefore the test case3 passed successfully.
Test Case 4	The user gives the inputin the form of result.	An output is predicted as the user gets a resultsimilar to the input image.	The result is that the user gets the result Therefore the test case4 passed Successfully.

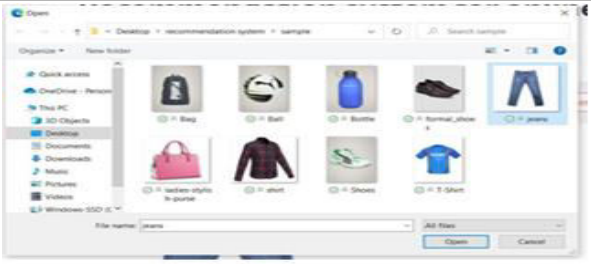

Fig 10: Now after selecting jeans as filename click on "Open". It will redirect you to the streamlet website.

8.1 OUTPUT

Deep Learning models for prediction recommendation products for the customers by extracting the features and using RESNET50 model and Image Net datasets. In this project we have used RESNET50 model which is a continuation of Convolution Neural Networks (CNN). RESNET50 has the ability to generate 2048 number of a dataset for every image that is to be processed. These dataset are called features. We also use Nearest Neighbor Search concept to analyze and display the most close and similar product that matches the given dataset.

Fig 8: Run the command "streamlet run main.py". It will open the streamlet website and run the website.

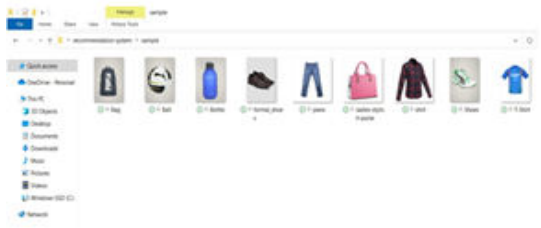

Fig 9: The set of sample images.

Fig 11: The output is shown 5 recommended Products for the given input image.

9. CONCLUSION

An intelligent clothing recommendation system based on visual similarity deep learning network running on Web platform was designed and implemented in this paper. After collecting existing clothing images from various databases, and applying a deep learning network to build a recommender system. The proposed system for recommending clothing apparel to users using a visual similarity approach also proves to diminish the cold start approach in other filtering approaches. After using a random image sample for testing, the system was appliedto give clothing recommendations, which had good performance (giving similarity scores above 80%) and reliability, and showed great potential in solving personalized recommendations.

Recommender systems open new opportunities of retrieving personalized information on the Internet. It also helps to alleviate the problem of information overload which is a very common phenomenon with information retrieval systems and enables users to have access to products and services which are not readily available to users on the system. Here we discussed the two traditional recommendation techniques and highlighted their strengths and challenges with diverse kind of hybridization strategies used to improve their performances.

10. FUTURE ENHANCEMENT

We believe that our proposed system will be able to help the users to choose from variety of products available while shopping from online platforms. It helps the users to select the best product of their choice. Our main goal is to provide users with various similar products which match their interest. In addition to this, any sort of contribution to this project will be beneficial to the users. Project can be updated in near future as and when requirement for the same arises, as it is very flexible in terms of expansion.

REFERENCES

*[1] PEARL PU, LI CHEN, RONG HU, (*October 2011), "A user centric evaluation framework for recommender systems",Proceedings of the fifth ACM conference on Recommender systems, https://doi.org/10.1145/2043932.2043962

[2] HESSEL TUINHOF, CLEMENS PIRKER, & MARKUS HALTMEIER, (July 17, 2018), "Image Based Fashion ProductRecommendation with Deep Learning", https://arxiv.org/pdf/1805.08694.pdf

[3] PIJUSH KANTI DUTTA PRAMANIK, AVICK KUMAR DEY & PRASENJIT CHOUDHURY, (January

2021), “Recommendersystems: an overview, research trends, and future directions”, International Journal of Business and SystemsResearch, https://www.resea.net

[4] SAAD ALBAWI; TAREQ ABED MOHAMMED; SAAD ALZAWI,(2017), “Understanding of a Convolution Neural Network”, International Conference on Engineering and Technology (ICET),10.1109/ICEngTechnol.2017.8308186

[5] MAHBUB HUSSAIN, JORDAN J. BIRD, AND DIEGO R. FARIA, (June 2018), “A Study on CNN Transfer *OLGA BELITSKAYA*, “Style Color Images”, https://www.kaggle.com/olgabelitskaya/style-colorimages

[6] ZIWEI LIU, PING LUO, SHI QIU, XIAOGANG WANG, XIAOOU TANG, (2016), “Deep Fashion: Powering Robust Clothes Recognition and Retrieval with Rich Annotations”, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 10.1109/CVPR.2016.124

[7] EKABA BISONG, (September 2019), “Building Machine Learning and Deep Learning Models on Google Cloud Platform”, Tensor Flow 2.0 and KERAS (pp 347-399), Springer, 10.1007/978-1-4842-4470-8_30

[8] F. O ZGENEL AND A. GO NEN SORGU, (2018), “Performance Comparison of Pertained Convolution Neural Networks on Crack Detection in Buildings”, 35th International Symposium on Automation and Robotics in Construction,

https://www.iaarc.org/publications/fulltext/ISARC2018-Paper154.pdf.