Contents

Implementation

Fashion Recommendation System

Overview

Fashion recommendation systems are a type of AI used in e-commerce to suggest clothes and outfits to users. They can be very helpful for people who are overwhelmed by choice, or who are looking for inspiration.The main purposes and goals of Fashion Recommendation systems include:

Purpose:

Fashion recommendation systems serve two main purposes: benefiting the **user** and the **retailer**.

User Benefits:

- **Enhanced Shopping Experience:** By filtering through vast options and suggesting relevant items, the system saves users time and effort. Instead of manually browsing through hundreds of products, users receive curated suggestions that match their preferences and needs.
- **Personalized Discoveries:** Recommendations go beyond just similar items, introducing users to new styles and trends they might be interested in. This personalization helps users discover products they might not have found on their own, broadening their fashion horizons.
- **Outfit Inspiration:** The system can help users put together complete outfits that complement each other, overcoming decision fatigue. Users can see how different items look together, making it easier to visualize and choose entire outfits.
- **Improved Confidence:** Getting recommendations based on user preferences can lead to users feeling more confident in their fashion choices. . Personalized suggestions ensure that users are more likely to find items they will love and feel comfortable wearing.

Retailer Benefits:

- **Increased Sales:** By suggesting relevant products, the system nudges users towards buying and can improve conversion rates. Personalized recommendations increase the likelihood of a purchase by showing users items that are tailored to their tastes.
- **Reduced Decision Overload:** When users find what they're looking for faster, they're more likely to stay on the platform and browse more. This reduced decision fatigue can lead to higher user engagement and longer browsing sessions.
- **Cross-Selling and Upselling:** Recommendations can introduce users to complementary items, like accessories or shoes, increasing their basket value. By suggesting additional products that go well with items already in the user's cart, retailers can boost overall sales.
- **Data-Driven Insights:** User data collected through recommendations helps retailers understand customer preferences and tailor their offerings. T

Applications:

Fashion recommendation systems have a variety of applications beyond just suggesting similar clothes or complete outfits. Here are some interesting ways they're being used:

- **Virtual Styling:** Imagine a system that can analyze your body type and suggest clothes that flatter your figure. This can be particularly helpful for online shopping where trying things on isn't possible.
- **Fit and Size Recommendations**: By integrating with virtual fitting rooms and using data from past purchases, these systems can recommend the best size and fit for each user, reducing the likelihood of returns.
- **Personalized Trend Exploration:** These systems can not only suggest trendy items but also tailor them to your individual style. So, you can discover new trends that fit your taste.
- **Style Evolution**: As users' tastes evolve, recommendation systems can adapt and continue to provide relevant trend suggestions, ensuring that users always have fresh, up-to-date options.
- **Occasion-Based Recommendations:** Need an outfit for a job interview or a night out? The system can analyze the occasion and recommend appropriate clothing choices.
- **Seasonal Recommendations**: The system can also provide suggestions based on the season or weather, ensuring users are both stylish and comfortable in their outfits.
- **Social Recommendation and Inspiration:** Imagine getting outfit recommendations based on what your friends or favorite influencers are wearing. This can be a great way to discover new styles and see how trends translate into real-world outfits.
- **Peer Reviews and Ratings**: Integrating peer reviews and ratings into the recommendation system helps users make informed decisions based on the experiences of others with similar tastes.
- **Sustainable Shopping:** Recommendation systems can be used to promote eco-friendly clothing brands or suggest ways to style pre-owned items, encouraging more sustainable fashion choices.
- **Eco-Friendly Brands**: Recommendation systems can promote eco-friendly clothing brands and suggest sustainable fashion choices. By highlighting brands that use ethical practices and sustainable materials, the system can help users make more environmentally conscious decisions.
- **Pre-Owned and Upcycled Fashion**: The system can suggest ways to style pre-owned items or upcycled fashion pieces, encouraging more sustainable shopping habits. This feature not only promotes sustainability but also helps users create unique, personalized looks.

These are just a few examples, and as technology advances, we can expect even more innovative applications for fashion recommendation systems in the future.

Integration of various recommendations

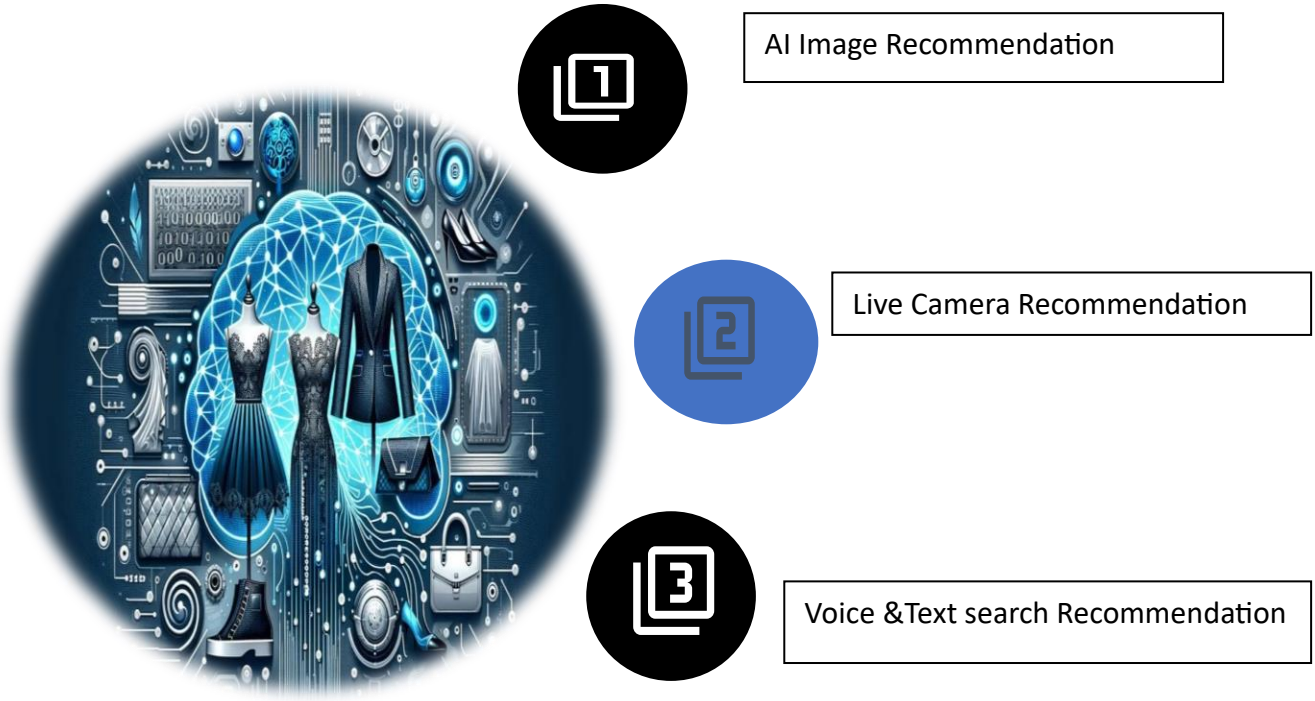


Image based recommendation :

Traditional recommendation systems rely on user history or explicit ratings. Image-based systems take a different approach. By analyzing the visual content of an image provided by the user, they aim to capture the user's style preferences and taste.

Colour based recommendation :

This technology aims to make putting together outfits a breeze. By analyzing the colors and potentially patterns in your clothing (or an object) through your camera, the system acts as a real-time stylist.

Voice & text search recommendation:

Voice-based clothing recommendations powered by text descriptions are an innovative way to streamline your shopping experience

Our Website interface:

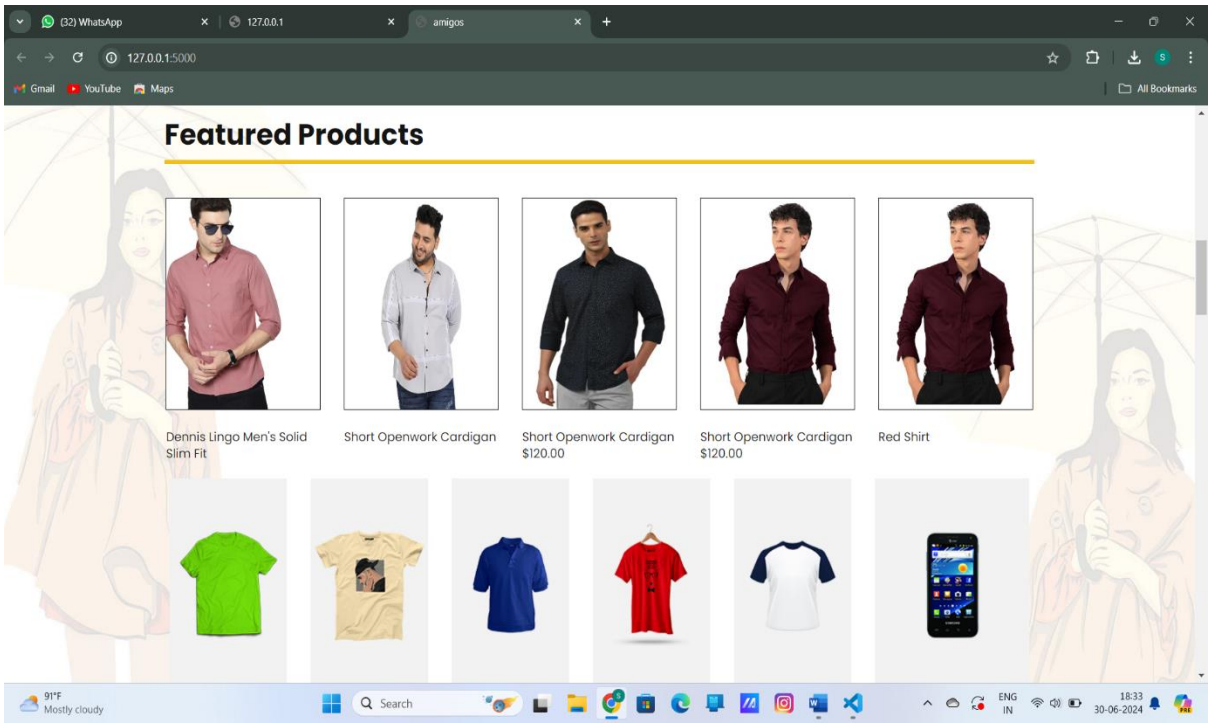


Image Based Recommendation

Traditional recommendation systems rely on user data like purchase history, browsing behavior, or explicit ratings. Image-based recommendation takes a different approach, focusing on the visual aspects of products to suggest similar items. This is particularly useful for products where aesthetics play a major role, like fashion or furniture.

Purpose and Applications:

Purpose of Image Based Recommendation:

- **Understanding User Preferences Through Visual Cues:** Traditional recommendation systems rely on user history or explicit ratings. Image-based systems take a different approach. By analyzing the visual content of an image provided by the user, they aim to capture the user's style preferences and taste. This goes beyond just similar products; it focuses on the aesthetics and visual characteristics the user is drawn to.

- **Recommending Similar Items Based on Visual Understanding:** Once the system understands the user's preferences through image analysis (colors, patterns, shapes, objects within the image), it recommends similar items. These recommendations go beyond simply similar products and delve deeper into capturing the user's visual style.

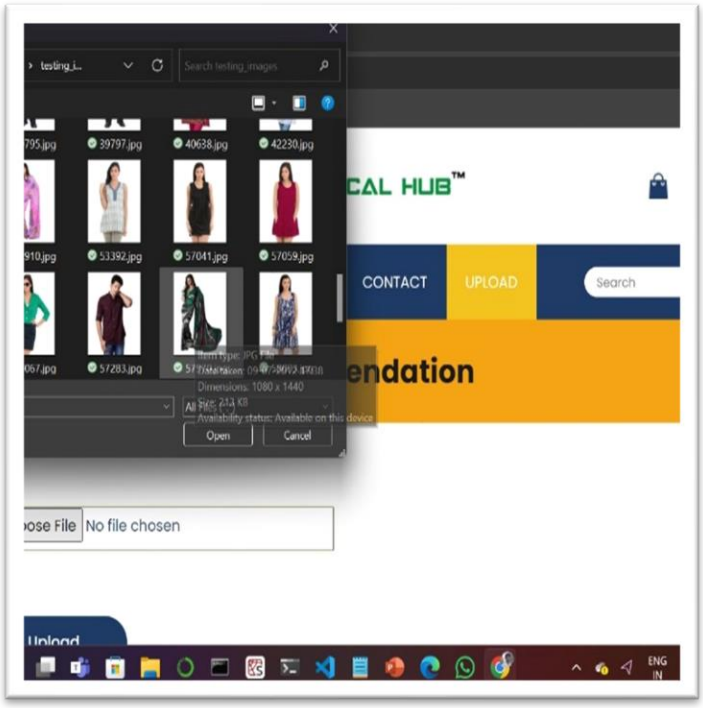
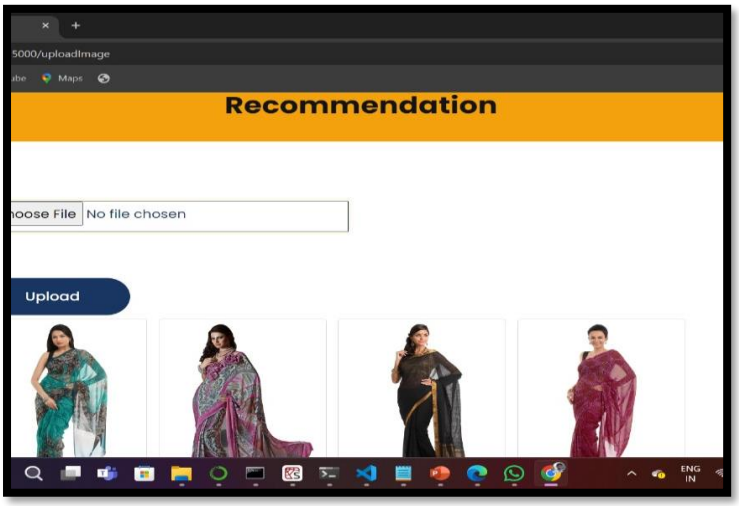


Image Recommandation

Recommanded products based on the Style and design



Applications of Image Based Recommendation:

- **E-commerce Platforms:** This is a prime area where image-based recommendations shine. Imagine browsing for clothes online and seeing a jacket you like. You can upload a picture of that jacket and the system recommends similar styles, colors, or patterns, even if they are from different brands. This is also applicable to furniture, home décor, or any product where aesthetics are a major buying factor.

- **Social Media Inspiration:** Social media platforms can leverage image-based recommendations to suggest products based on user-uploaded photos. For instance, you could post a picture of your new shoes and the platform recommends similar styles or complementary accessories.

- **Personalized Trend Exploration:** These systems can analyze user-uploaded photos and suggest trendy items that fit their individual style. This allows users to discover new trends while staying true to their preferences.

- **Virtual Styling and Design:** Image-based recommendations can be integrated into virtual styling tools. By uploading a picture of yourself, the system could recommend clothes that flatter your body type or suggest how to style different pieces together. Similarly, for interior design tools, users could upload an image of their room and get recommendations for furniture or décor that matches the existing style.

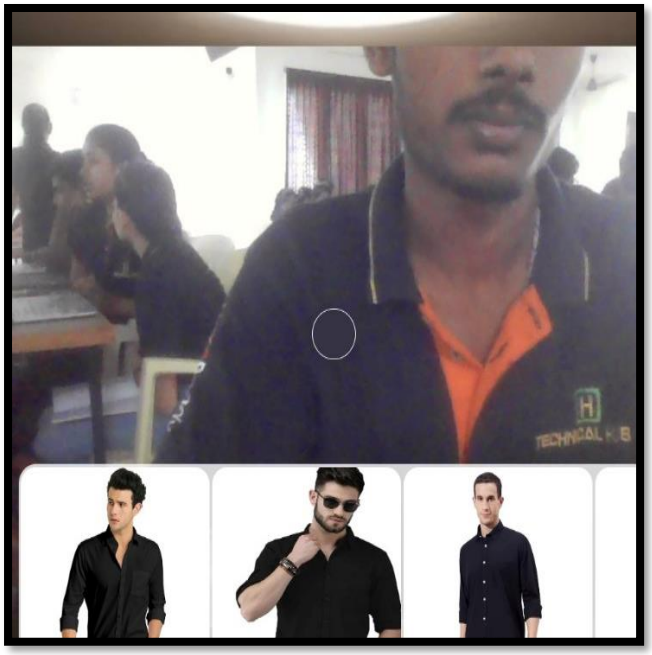
Colour Based Recommendation:

Imagine stepping into a virtual closet where clothes magically appear based on what you wear! Livecamera color detection for clothing recommendations is transforming the fashion world. This innovative technology bridges the gap between your style inspiration and real-time outfit suggestions.

Purpose:

- **Simplifying Outfit Creation:** This technology aims to make putting together outfits a breeze. By analyzing the colors and potentially patterns in your clothing (or an object) through your camera, the system acts as a real-time stylist. It suggests complementary pieces like bottoms, accessories, or shoes to create a cohesive look. This eliminates the time and effort spent browsing endless options and struggling to match colors.

- **Personalized Style Inspiration:** Live camera color detection goes beyond basic recommendations. It can spark new ideas and help you discover trends. Imagine seeing a stylish color combination on someone or in your environment. Simply point your camera and get instant suggestions for similar clothing items that you can incorporate into your wardrobe. This fosters a more personalized and inspiring way to approach fashion.



Live Video Recommandation

Based on the colour Products will be recommanded

Applications:

- **Effortless Outfit Creation:** This is the core application. Feeling overwhelmed by your closet? Point your camera at a top or a piece of clothing you already own, and the system recommends complementary items to complete your outfit. It can suggest bottoms, jackets, shoes, accessories, or even jewelry that create a cohesive look based on color, pattern, or style.

- **Seamless Shopping Experience:** Imagine browsing online stores where clothes magically appear in color combinations that complement your existing wardrobe. No more endless scrolling through clothes that might not match what you already own! This technology can analyze a piece from your closet through your camera and suggest similar items or outfits you can purchase online.

- **Trend Inspiration on the Go:** See a captivating color combination on the street or in a magazine? Capture it with your camera and get instant recommendations for similar clothing items. This is a great way to stay updated on the latest trends and discover new outfit ideas based on what catches your eye in real-time.

- **Personalized Wardrobe Planning:** Planning outfits for a trip or a special occasion? Use the system to experiment with different color combinations and styles based on the clothes you already have. You can virtually create multiple outfit options and see how they look together before making packing or purchasing decisions.

- **Coordinated Group Outfits:** Planning outfits for a group event or photo shoot? This technology can be a game-changer. . Everyone can use their cameras to analyze their chosen pieces, and the system can recommend complementary outfits for the entire group, ensuring a coordinated and stylish look.

- **Accessibility Tool for Colorblind Users:** For people with color blindness, choosing complementary colors can be challenging. This system can act as an assistive tool, analyzing colors through the camera and suggesting outfits that work well together based on accessibility guidelines.

Voice & Text Based Recommendation:

Voice-based clothing recommendations powered by text descriptions are an innovative way to streamline your shopping experience. Imagine browsing through outfits that perfectly match your style, all through the power of your voice! This technology offers a unique blend of convenience and personalization, making it easier than ever to find clothes you love.

Purpose:

The purpose of a voice text-based clothing recommendation system is two-fold:

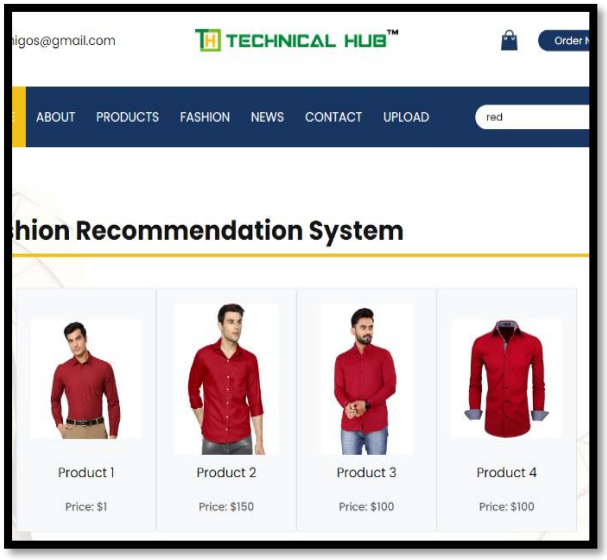
1. **Break Down Text Barriers:**

   This system bypasses the need for users to type detailed descriptions of their desired outfit. By allowing users to speak their style preferences, it caters to those who might find text input cumbersome or struggle to articulate their fashion desires in writing.

2. **Enhance Personalization:**

   By using natural language processing, the system can understand the nuances of spoken language. It can pick up on keywords, emotions (wanting to feel confident, playful, etc.), and even stylistic intonations (formal vs casual speech) to create a more personalized clothing recommendation.

Text & voice search recommendation



Recommanded products based on the text and voice

Applications:

Here are some applications for voice text-based clothing recommendation systems:

- **Effortless Outfit Brainstorming:** Stuck in a style rut or unsure what to wear for an upcoming event? Simply describe your desired look (e.g. "something comfy for a brunch date" or "a dressy outfit for a work presentation") and let the system generate outfit ideas based on your voice description.
    -
- **Accessibility for People with Visual Impairments:** This system can be a valuable tool for users with visual impairments. They can describe their desired style, color preferences, or outfit formality, and the system can recommend suitable clothing options through voice interaction.
-
- **Multitasking Fashion Exploration:** Imagine browsing clothes while doing chores or commuting! Describe the type of clothes you're looking for while multitasking, and the system can recommend options you can explore later. This is a convenient way to stay up-to-date on trends or casually search for new pieces while on the go.
-
- **Refine Recommendations with Iterations:** Unlike a static text search, voice interaction allows for back-and-forth communication. You can describe your initial preferences, and then refine the search based on the recommendations provided by the system. For example, you could say "I like this style, but maybe in a different color."
-
- **Personalized Shopping Assistant:** This technology can evolve into a virtual shopping assistant. Describe a specific clothing item you have in mind (e.g. "a flowy maxi dress with floral prints"), and the system can search online stores and recommend similar options that fit your voice description.

Implementation Of Fashion Recommendation System(FRS)

Models and Libraries Involved

Data Sources:

- **User Data:**

  Purchase history, browsing behavior, saved items, ratings, and voice descriptions (in text-based systems).

- **Item Data:**

  Information about the clothes themselves, like color, style, brand, material, size, and even image data.

- **External Data:**

  Fashion trends, social media data (what's popular on influencers' pages), and even weather information can be used to tailor recommendations.

Machine Learning Models:

- **Collaborative Filtering:**

  This technique analyzes user-user interactions to recommend items that users with similar tastes have purchased or liked. (Libraries: Scikit-learn (Python), Apache Mahout (Java))

- **Content-Based Filtering:**

  This approach focuses on item attributes like color, style, and brand to recommend similar items to those the user has interacted with. (Libraries: TensorFlow (Python), PyTorch (Python))

- **Matrix Factorization:**

  This technique breaks down user-item interaction data into a lower-dimensional space, capturing latent factors that influence user preferences. (Libraries: scikit-learn (Python), Surprise (Python))

Deep Learning Models:

- **Natural Language Processing (NLP) Models:** For voice-based systems, NLP models are used to understand the user's spoken descriptions, extract keywords and sentiment, and translate them into actionable data for recommendations. (Libraries: spaCy (Python), NLTK (Python))

- **Convolutional Neural Networks (CNNs):**

  Inception V3 is a pre-trained CNN architecture specifically designed for image recognition. It can be a powerful tool in fashion recommendation systems. By feeding product images into the Inception V3 model, the system can extract high-level features like patterns, textures, and styles. These features can then be used to recommend similar clothing items to the user based on their uploaded image or browsing history. (Libraries: TensorFlow (Python), PyTorch (Python))

Additional Points about Inception V3:

- **Transfer Learning:**

  Inception V3 is often used in a technique called transfer learning. The pre-trained model weights are leveraged as a starting point, and then fine-tuned with a specific fashion image dataset to improve its accuracy in recognizing clothing styles and features.

- **Benefits:** Inception V3 offers several advantages:
  - **High Accuracy:** It's been shown to achieve good performance on image recognition tasks.
  - **Efficiency:** By using a pre-trained model, developers can save time and resources compared to training a CNN from scratch.
  - **Scalability:** The model can handle large datasets of clothing images effectively.

OpenCV

OpenCV (Open Source Computer Vision Library) is an open-source computer vision and machine learning software library. It includes several hundred computer vision algorithms.

- **Purpose in the Code:**
  - **cv2.dnn.readNet**: Loads a pre-trained deep neural network model.
  - **cv2.dnn_DetectionModel**: Creates a detection model from the loaded network.
  - **cv2.VideoCapture**: Captures video from a camera or video file.
  - **cv2.imshow**: Displays an image or video frame.
  - **cv2.flip**: Flips an image or video frame.
  - **cv2.putText**: Puts text on an image or video frame.
  - **cv2.rectangle**: Draws rectangles on an image or video frame.
  - **cv2.imencode**: Encodes an image into a specific format (e.g., JPEG).

- **Data Source for FRS :**
  https://www.kaggle.com/datasets/paramaggarwal/fashion-product-images-dataset/code

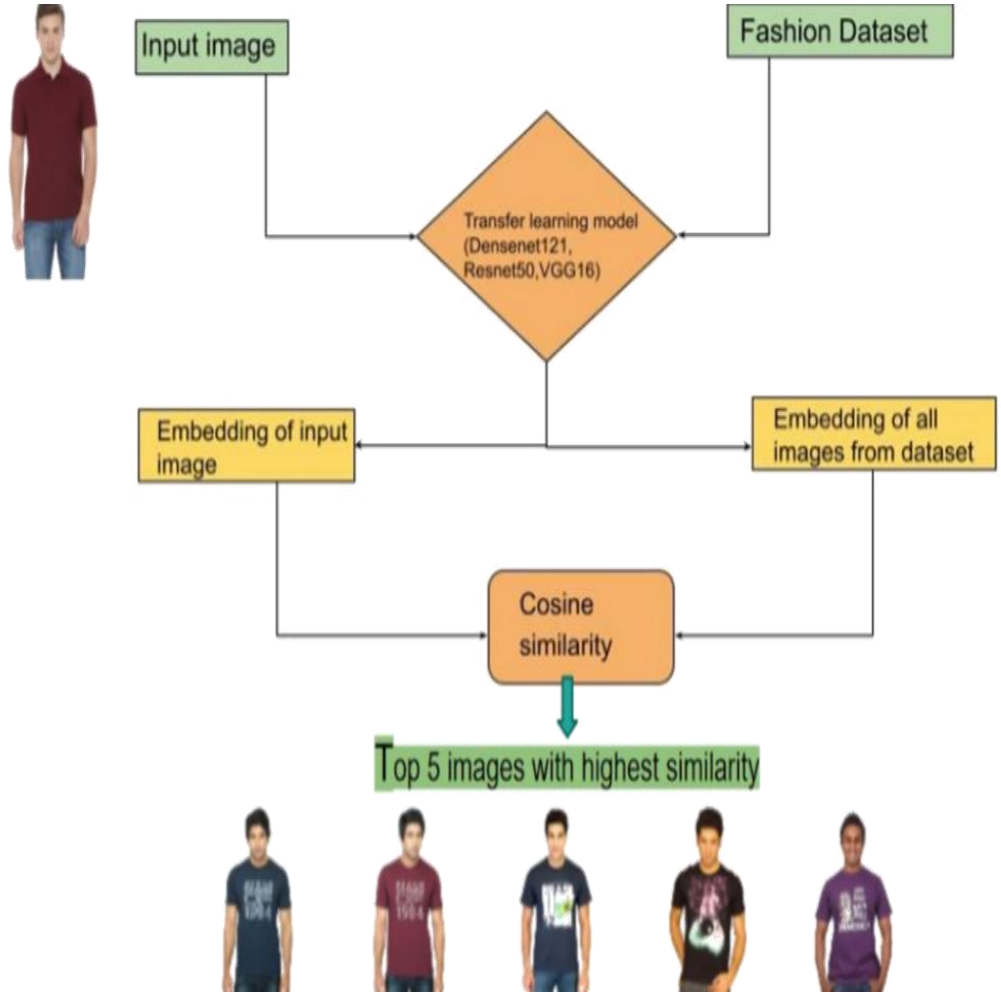
---

Code for Fashion Recommendation System:

```
from zipfile import ZipFile
import os
import glob
from tensorflow.keras.preprocessing import image as keras_image
from tensorflow.keras.applications.inception_v3 import InceptionV3, preprocess_input
from tensorflow.keras.layers import GlobalAveragePooling2D ,Dense
from tensorflow.keras.models import Model
import numpy as np
from PIL import Image
import matplotlib.pyplot as plt
from scipy.spatial.distance import cosine
from tensorflow.keras import layers, Sequential

from tensorflow.keras.models import Model
from tensorflow.keras.layers import Dense, GlobalAveragePooling2D


from tensorflow.keras import Sequential
from tensorflow.keras import layers
from tensorflow.keras.preprocessing import image as keras_image
import numpy as np
from tensorflow.keras.applications.inception_v3 import InceptionV3, preprocess_input

from keras.models import load_model
savedModel=load_model(r'E:\dp\project_space_code\finalweights.h5')

import pickle

with open('names.pkl','rb') as f: all_image_names = pickle.load(f)
with open('features.pkl','rb') as f: all_features = pickle.load(f)
import glob
# Correctly generate the list of image file paths
image_directory = r"D:\images"
image_paths_list = glob.glob(os.path.join(image_directory, '*'))



def preprocess_image(img_path):
    img = keras_image.load_img(img_path, target_size=(224, 224))
    img_array = keras_image.img_to_array(img)
    img_array_expanded = np.expand_dims(img_array, axis=0)
    return preprocess_input(img_array_expanded)

def extract_features(model, preprocessed_img):
    features = model.predict(preprocessed_img)
```

```
    flattened_features = features.flatten()
    normalized_features = flattened_features / np.linalg.norm(flattened_features)
    return normalized_features


def recommend_fashion(input_image_path, all_features, all_image_names, model, top_n=5):
    preprocessed_img = preprocess_image(input_image_path)
    input_features = extract_features(model, preprocessed_img)
    print(input_features,type(input_features))

    similarities = [1 - cosine(input_features, feature) for feature in all_features]
    top_indices = np.argsort(similarities)[-top_n-1:]

    # Ensure the subplot grid is dynamically sized to accommodate all images
    num_images = top_n + 1  # Number of images to display includes the input image and top N recommendations
    plt.figure(figsize=(20, 10))

    # Display the input image
    plt.subplot(1, num_images, 1)
    plt.imshow(Image.open(input_image_path))
    plt.title("Input Image")
    plt.axis('off')

    # Adjust the loop to correctly handle subplot positions
    rec_position = 2  # Start placing recommendation images from the second position
    for idx in top_indices:
        if all_image_names[idx] == os.path.basename(input_image_path):
            continue  # Skip the input image

        plt.subplot(1, num_images, rec_position)
        plt.imshow(Image.open(image_paths_list[idx]))
        plt.title(f"Rec {rec_position-1}")
        plt.axis('off')
        rec_position += 1

        # Stop if we've filled up the intended number of recommendations
        if rec_position > top_n + 1:
            break

    plt.tight_layout()
    plt.show()
input_image_path = r"C:\Users\saiku\OneDrive\Desktop\testing_images\28542.jpg"
# Adjust to an actual file path
recommend_fashion(input_image_path, all_features, all_image_names, savedModel, top_n=5)
```

Our Backend Process:



Input Image: This is the starting point of the process. Any product-based image is provided as input to the system

Deep Learning Model: The preprocessed image is fed into a deep learning model that has been trained on product images. This model could be a convolutional neural network (CNN) or any other architecture suitable for image recognition tasks.

Feature Extraction: The model extracts features from the input image. These features represent various characteristics of the product such as color, texture, shape, etc.

Similarity Calculation: The extracted features are compared with a database of product features. This comparison is usually done using similarity metrics such as cosine similarity or Euclidean distance

Matching Products: Based on the similarity scores, the system identifies products from the database that closely resemble the input image.

Output: The matched products are presented as output, either in the form of a list or visually represented to the user

Implementation Of Flask:

Libraries and Models involved:

Libraries used:

Flask

Flask is a micro web framework written in Python. It is lightweight and easy to use, making it a popular choice for web development, particularly for small to medium-sized applications.

- **Purpose in the Code:**
  - **Flask**: Creates the Flask application instance.
  - **render_template**: Renders HTML templates.
  - **Response**: Generates HTTP responses.
  - **url_for and redirect**: (Not used in the current code, but commonly used for URL handling and redirects.)

Code implementation of Flask:

```
from flask import Flask, request, redirect, url_for, render_template

import frs, pickle, os

from keras.models import load_model



app = Flask(_name_)



# Specify the directory to save uploaded files

UPLOAD_FOLDER = 'uploads'

app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER



@app.route('/')

def home():

   return render_template('index.html')



@app.route('/about')
```

```
def about():
    return render_template('about.html')


@app.route('/products')
def products():
    return render_template('products.html')


@app.route('/fashion')
def fashion():
    return render_template('fashion.html')


@app.route('/news')
def news():
    return render_template('news.html')


@app.route('/contact')
def contact():
    return render_template('contact.html')


@app.route('/upload')
def upload():
    return render_template('upload.html')



# Shirts pages
```

```
@app.route('/red')
def red():
    # return ("red shirts page")
    return render_template('red.html')

@app.route('/green')
def green():
    # return ("green shirts page")
    return render_template('green.html')

@app.route('/kurtas')
def kurtas():
    # return (" kurtas page")
    return render_template('kurtas.html')

@app.route('/sarees')
def sarees():
    # return (" sarees page")
    return render_template('sarees.html')


@app.route('/yellow')
def yellow():
    # return (" sarees page")
    return render_template('yellow.html')
```

```
@app.route('/black')
def black():
    # return (" sarees page")
    return render_template('black.html')


@app.route('/purpleshirts')
def purpleshirts():
    # return (" sarees page")
    return render_template('purpleshirts.html')


@app.route('/blue')
def blue():
    # return (" sarees page")
    return render_template('blue.html')


@app.route('/casual')
def  casual():
    # return (" sarees page")
    return render_template('casual.html')
```

```
@app.route('/cameras')
def cameras():
    # return (" sarees page")
    return render_template('cameras.html')


# @app.route('/uploadImage', methods=['POST'])
# def upload_file():
#     if 'file' not in request.files:
#         return redirect(request.url)
#     file = request.files['file']
#     if file.filename == '':
#         return redirect(request.url)
#     if file:
#         filename = file.filename
#         file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
#         return 'File uploaded successfully'


@app.route('/uploadImage',methods=["GET","POST"])
def uploadImage():
    file=request.files['file']
    file_path = 'static/uploads/' + file.filename
    file.save(file_path)


    # from  pkl files (features.pkl, names.pkl)
    with open('names.pkl','rb') as f: all_image_names = pickle.load(f)
```

```
    with open('features.pkl','rb') as f: all_features = pickle.load(f)

    # save model

    savedModel=load_model(r'D:\finalweights.h5')



    input_image_path = file_path

    # Adjust to an actual file path

    k = frs.recommend_fashion(input_image_path, all_features, all_image_names, savedModel,
top_n=4)



    t=[]

    for i in k:

        t.append(i.replace("D:\\images\\", ''))

    print(t)



    # t=[]

    # for i in k:

    #     file_path = 'static/recommendations/' + i[13:]

    #     t.append(i[13:])

    #     file.save(file_path)



    return render_template('upload.html',res=t)



if _name_ == '_main_':

    app.run(port=5000, debug=True)
```

Advantages of fashion recommendation system:

Time-saving:

Personalized recommendations reduce browsing time, allowing users to swiftly discover items matching their style, thus streamlining decision-making.

Enhanced Shopping Experience:

Tailored selections cater to users' tastes, making shopping not just efficient but also enjoyable and satisfying.

Discover New Styles:

Recommendations expose users to fresh trends and brands, broadening their fashion repertoire and inspiring creativity in wardrobe choices.

Simplified Shopping:

Personalized suggestions eliminate the hassle of shifting through options, enabling users to quickly find items that suit preferences and budget, enhancing overall convenience

Conclusion:

- The future of fashion recommendation systems looks really promising. These systems are getting better at suggesting clothes and accessories that suit your style and preferences. We use AI to do this. So, get ready for a whole new way of shopping

- We're currently in the planning phase for our website launch! We're brainstorming ideas and laying out our vision to create an engaging and user-friendly online platform. Our goal is to offer a seamless shopping experience that caters to your needs and preferences. Stay tuned for more updates as we work diligently to bring our website to life!