VIRTUAL-TRY-ON (VITON) OF UPPER BODY FASHION ACCESSORIES FOR INDIAN POPULATION

Capstone Project Proposal by Team -7

Capstone Project Proposal

- Title
- Brief Problem Statement
- Background information
- Motivation for selection of the project
- Detailed dataset description and dataset source
    - The Instagram collection
    - The YouTube collection
    - Cleaning up the dataset
    - Creating slices of users from a photo Code
        - block
        - Sample Output
    - The train dataset
- Current benchmark: provide references (if any)
    - Recent announcements in the Fashion TryOn space Methodology
    - Algorithms/Models
    - Metrics
- Proposed Plan
    - Approaches
    - Annotating the key points and accessory location in the dataset
    - Analyzing the gaze in the user photo
        - Code block
        - Sample output
    - Stages with defined deliverables
- Preliminary Exploratory data analysis
- Expected outcomes
- Project demonstration strategy
- Proposed timeline of project stage executions
- Team members' names
- References

Title

Building a Virtual-try-On (VITON) of upper body fashion accessories (jewellery) for Indian population.

Brief Problem Statement

Allow the user to upload an image and then let the user to try an in-shop accessory on the user's photo. In particular, we can think of it as a subclass of Style transfer problem.

Background information

Broadly speaking, the realm of intelligent fashion can be divided into four categories.

1. Fashion detection includes landmark detection, fashion parsing, and item retrieval
2. Fashion analysis contains attribute recognition, style learning, and popularity prediction
3. Fashion synthesis involves style transfer, pose transformation, and physical simulation
4. Fashion recommendation comprises fashion compatibility w.r.t Jewellary.

This project falls into the subclass of Fashion synthesis, namely, style transfer. Style transfer is transferring an input image into a corresponding output image such as transferring a real-world image into a cartoon-style image, transferring a non-makeup facial image into a makeup facial image, or transferring the clothing, which is tried on the human image, from one style to another. Broadly, style transfer is categorized into two subclasses:

1. Facial makeup
2. Virtual try-on (VITON).

Here we are focusing on the latter (see Figure-1).

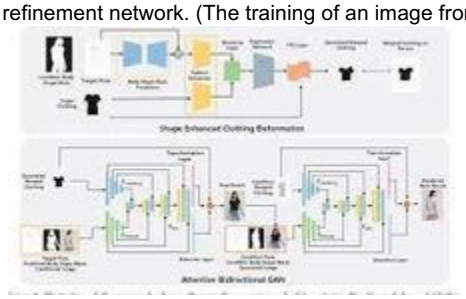
Motivation for selection of the project

Virtual Try-on feature has revolutionized customers buying experience. The pandemic has forced adoption of digital technology, changing customer behavior to buy items online. Online, it is hard to know if the accessory fits people or looks good on them, so customers naturally gravitate towards sites where they can virtually try on the piece of accessory. Hence, Virtual Try-on, which was previously good to have feature, now has become a must have feature for the merchants to boost their sales and customers to experience what they are buying. Hence, we are attempting to build this for Indian accessories which can be later used by small and medium size merchants on their Instagram or Meta pages.

Initial Implementation Plan

The idea begins with this work where they've described the VITON framework (which is basically a GAN architecture). The goal of VITON is, given a reference image "**I**" with a clothed person and a target clothing item to synthesize a new image, where "**c**" is transferred naturally onto the corresponding region of the same person whose body parts and pose information are preserved.

In a practical virtual try-on scenario, only a reference image and a desired product image are available at test time. They adopt the same setting for training, where a reference image "**I**", with a person wearing "**c**" and the product image of "**c**" are given as inputs. Now the problem becomes given the product image c and the person's information, how to learn a generator that not only produces "**I**" during training, but more importantly is able to generalize at test time – synthesizing a perceptually convincing image with an arbitrary desired clothing item. To this end, the authors introduced a clothing-agnostic person representation of the image. The reference image is synthesized with an encoder-decoder architecture conditioned on the person representation as well as the target clothing image. The resulting coarse result is further improved to account for detailed visual patterns and deformations with a refinement network. (The training of an image from this procedure is given in Figure-2)



However, VITON fails to handle large deformation, especially with more texture details, due to the imperfect shape-context matching for aligning clothes and body shape. Various modifications of VITON were proposed to address this problem. Most notably, the characteristic-preserving virtual-try-on (CP-VTON), feature-preserving virtual-try-on (VTNFP), shape-preserving virtual-try on (SP-VITON), Conditional Analogy Generative Adversarial Network (CAGAN), adaptive content generating and preserving network (ACGPN) etc. were proposed. A comparison between VITON and CP-VTON is given in Figure-3.

Different from the previous works that needed the in-shop clothing image for virtual try-on, given an input image and a sentence describing a different outfit FashionGAN was designed to "redress" the person. Another related work, namely, M2E-TON was designed to try on clothing from humanA image to humanB image, when they are in different poses.

For this project, we adopt the fashion transfer model proposed in the FashionOn project. The dataset for this project is available here. We will discuss the details of the FashionOn architecture in the methodology section. With respect to the Fashion transfer models, FashionOn tackles the following three issues:

- How to properly deform the new clothing item and seamlessly align with the target person
- How to generate the try-on image that maintains not only the detailed visual features of the clothing item, like the texture and color, but also the other body parts of the person, while changing the person pose
- There is no large-scale benchmark dataset that can support the research of our new virtual try-on task.

Pivot from Initial Implementation

Achieving the established goal by following the initial implementation plan in the stipulated time-frame was a humongous task. Our mentor's timely feedback motivated us to look for simpler and easier solutions to tackle the same problem. More research indicated easier solutions to the problem using OpenCV by doing facial key-point detection and placement of jewelry on the detected key-points. Hence, we gravitated towards exploring multiple pretrained models like Mobilenet, YOLO, DLIB, movenet, media-pipe, blazeface. Different models give different facial landmark points. In the final solution, each of the jewelry categories uses the model giving the most near-accurate landmark to place the jewelry.

We intend to explore the intended initial implementation using GAN's again in future.

Dataset Creation

Right from the onset, the biggest challenge was the unavailability of dataset for images with Indian Faces. Unavailability of the jewelry dataset was the next big challenge. The initial part of our project journey was dedicated to resolving these two problems by downloading and collecting images, filtering the relevant images, post processing, annotating and creating a dataset and extracting meaningful data that could be fed to train the models.

A significant volume of our effort and work went into collecting images, cleaning them, annotating images and then train the models with these images.
.

Detailed dataset description and dataset source

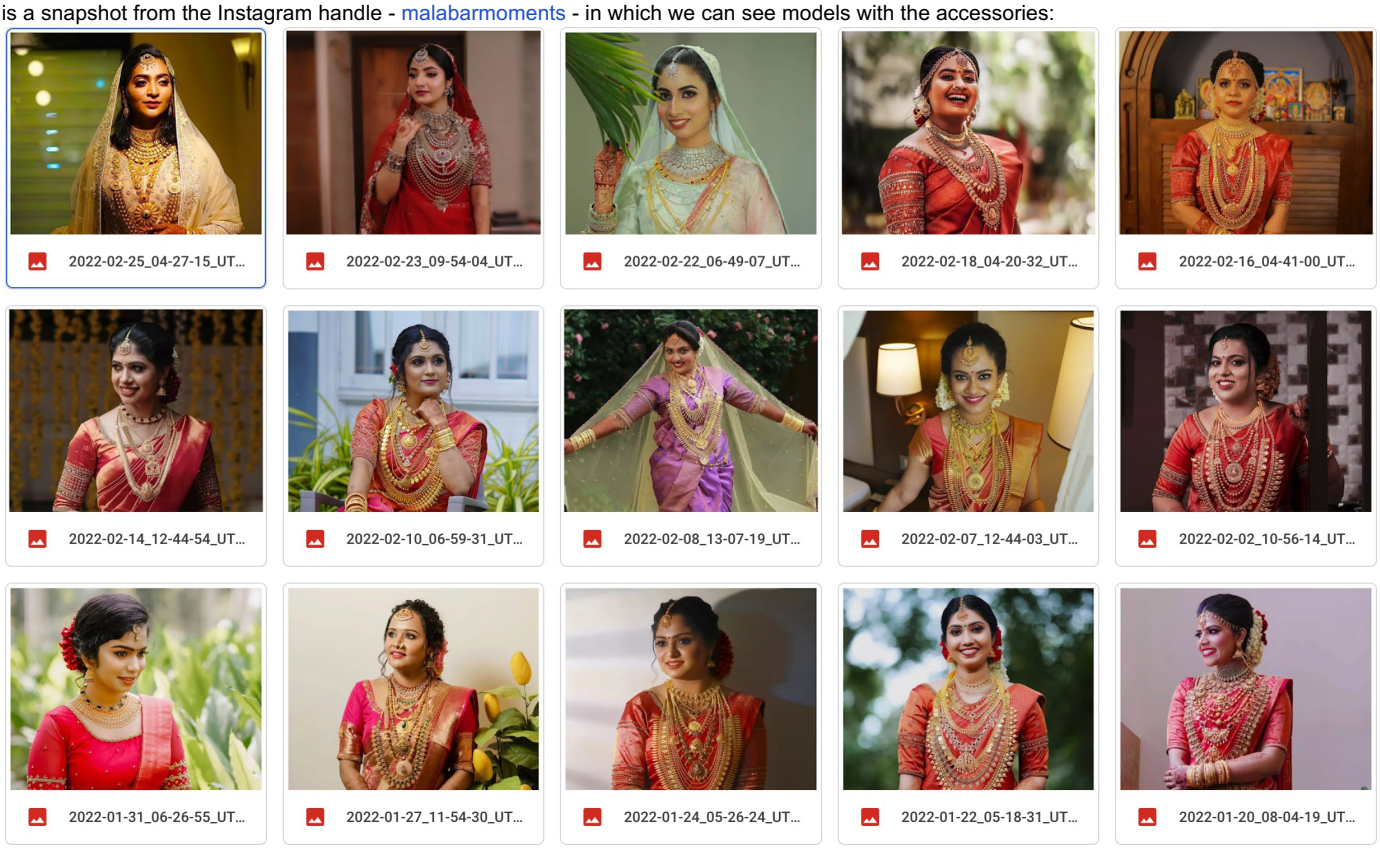
The original VITON dataset contains pairs of frontal-view women and top clothing images, however, that is now restricted due to copyright laws. However, VITON dataset can be obtained from the ACGPN dataset. Furthermore, we also have DeepFashion dataset which contains upper-body images, sentence descriptions, and human body segmentation maps. Our dataset will be similar to the dataset for the FashionOn project.
Broadly, FashionOn dataset contains pairs of same person with same clothing in 2 different poses and one corresponding clothing image. Our dataset will contain pairs of the same person with same fashion accessory in 2 different poses and one corresponding accessory image.

Since, there are no datasets for the accessories for Indian population we need to create the dataset from scratch. We are relying on Instagram handles that sell fashion accessories for Indians (see this, for example) and promotional videos of jewellery products that are put on YouTube (see this, for example).

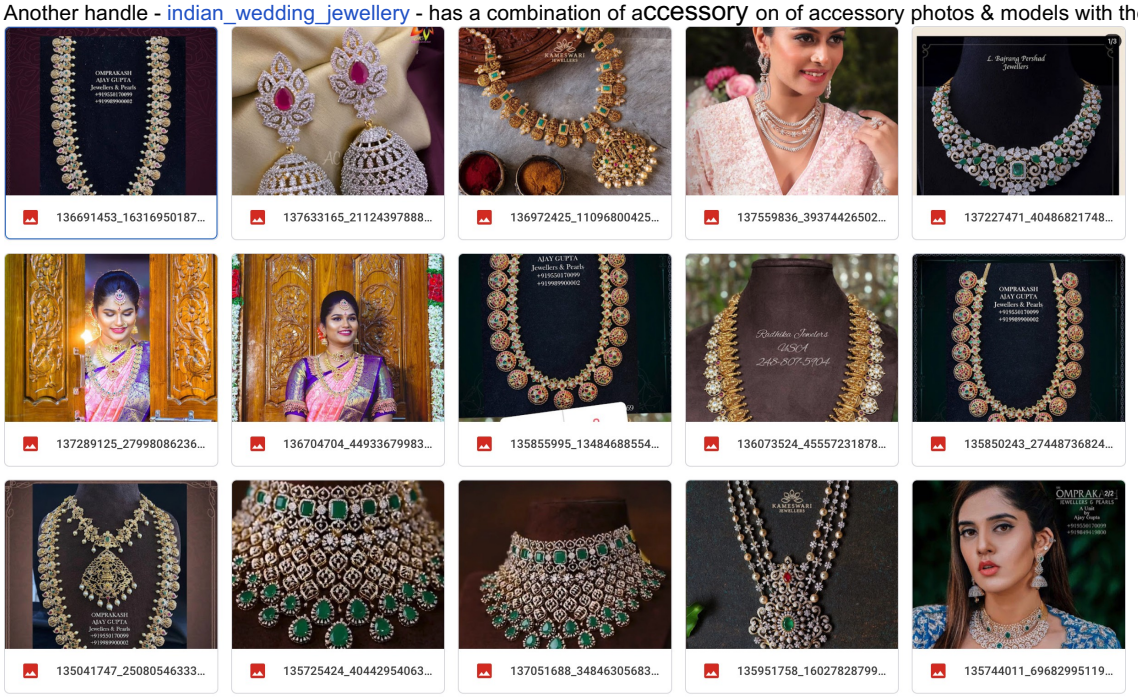
The Instagram collection

In Instagram, a handle may have posted just the fashion accessory (or) a muse modeling with the fashion accessory (or) a video. Here

is a snapshot from the Instagram handle - malabarmoments - in which we can see models with the accessories:



Another handle - indian_wedding_jewellery - has a combination of accessory on of accessory photos & models with these accessories.



The YouTube collection

Promotional videos from jewellery merchants are a big source of images. The video frames could end up being doubly useful to help generating images of the person without the accessory and with the accessory. To explain this with an example, look at the below images taken from this video:

Cleaning up the dataset

Given the wide variety of this downloaded content from sources like Instagram, YouTube, we have filtered out the images and picked only those which have a human with a fashion accessory. This was immensely helpful to arrive at an understanding of how and where to place these accessories by a deep learning model.

Creating slices of users from a photo

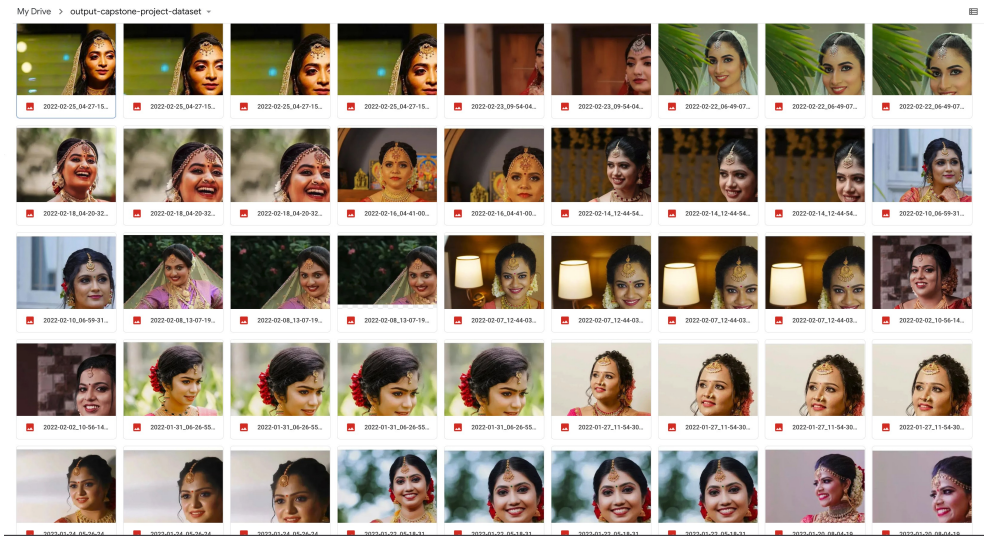
We used a tweaked version of the python package - face_recognition, for creating photo slices (that include the jewellery) from a larger photo (single / group)

Code block for photo slices

```
files_list = os.listdir(DATASET_PATH)
for photo_file in files_list:
  full_image_path = DATASET_PATH + photo_file
  try:
    if '.' in full_image_path:
      get_slice_from_image(file_name=full_image_path)
    else:
      print(f"{photo_file} is not an image")
  except Exception as ex:
    pass
```

Sample Output

The first image below shows a collection of our initial dataset - the images downloaded from Instagram (or) picked from videos. The second image below shows the slices from the above codeblock.



The Train Dataset

For a given accessory, we will need 2 or more variants of a model wearing this accessory. Explaining this with a sample:

Annotations

Annotation was done in 2 ways. Through Dlib and by manual annotaions.We did manual annotation for 695 images and through Dlib,its 394.

Among those 61% was frontal face images and 39% was side images.The entire population of annotation was done mainly on imgaes of indian females consisting of 80% of the images and male and kids in the ration 18% and 2% respectively. Out dataset was consisting of 83% of images from the web and 17% images from the teammates.

- **Annotating the key points and accessory location in the dataset**

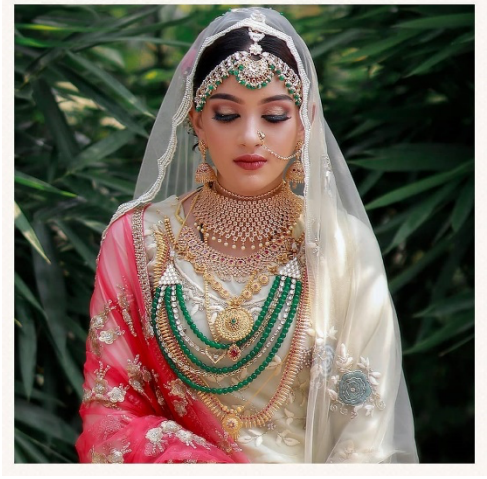
Given the greenfield that we are operating in, the need to have high levels of annotation for the dataset. We will be taking the image-slices we created and annotating them with the highly popular VGG Image Annotator tool. These annotations can also be validated and augmented using mediapipe. A sample mediapipe for our use-use-case looks like the image below:

- **Manual Annotations**

Annotations were done on 800 images in the exact same order!! Since multiple people were involved in annotations, maintaining consistency was the biggest challenge. We used VIA Annotation tool forannotating our images where we kept few parameters commaon across the team to maintain consistency.

Index	Name	Name Identifier
0	Forehead Center	Forehead
1	Left Ear	left_ear
2	Right Ear	right_ear
3	Left Nostril	left_nose
4	Right Nostril	right_nose
5	Left Neck top point	left_neck_top_point
6	Left Neck Bottom Point	left_neck_bottom_point
7	Right Neck Top Point	right_neck_top_point
8	Right Neck Bottom Point	right_neck_bottom_point
9	Neck lower Center point	neck_center

Input:



Output:

Analyzing the gaze in the user photo

We used the headpose_final package for analyzing the gaze in the user photo. We believe that the gaze will help increasing the accuracy of placing the jewellery in the user photo.

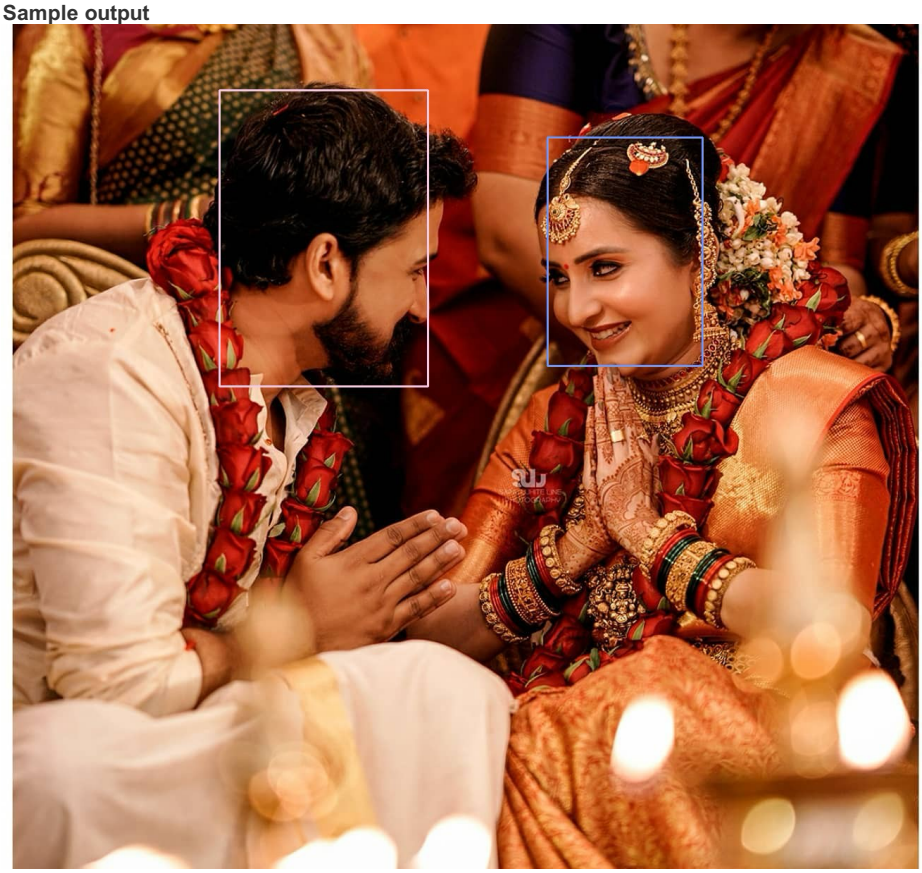
Code block for analyzing the Gaze

```
from ln_tracking_heads import

load_detection_model ### Detection
# Load the SSD detector
model = load_detection_model()

image = cv2.imread('/content/test-image-
multiple.jpg') try:
  dets = detect_on_image(image, model,
  verbose=True) print(dets)
except
  Exception
  as ex:
  pass
```

Sample output



Stages with defined deliverables

1. A service that will return the list of available fashion accessories sourced from various sellers. On the longer run (in a full-blown consumer facing product), we want this microservice to be the aggregator of all fashion accessories from various sellers, which is retrieved through a web-hook (or) can be fetched from a Content Management System.
2. A service that will take the user's choice of fashion accessory and process the user uploaded photo
3. A service that will place the selected accessory in the user's photo and return the image that will be presented to the user

Preliminary Exploratory data analysis

The solution space is to support feature for plain gold Nose-ring, Ear-ring, Maang Tika, Necklace and Chain for Indian Females.

So, for each item category - Nose-ring, Ear-ring, Maang Tika, necklace and chain, three set of images were collected. Each set had the item image, model with item in pose 1 and model with item in pose 2. The plan was to collect 1000 such sets for each category. In addition, there were annotations data to highlight the accessory, part where it was worn.

Expected outcomes

Virtual Try-On for an accessory - Ability for a user to try the accessory on given image by uploading her picture.

Project demonstration strategy

We will deploy the web-app under a product brand name and share the URL with the demo audience. When we have the Alpha version of our app, we are planning to invite friends and family to try the product and give feedback. This will help us refine the UX of the product & also measure the model accuracy.

Executed timeline of project stage executions

Include weekly progress goals for each of the 4 Capstone Project mentored sessions

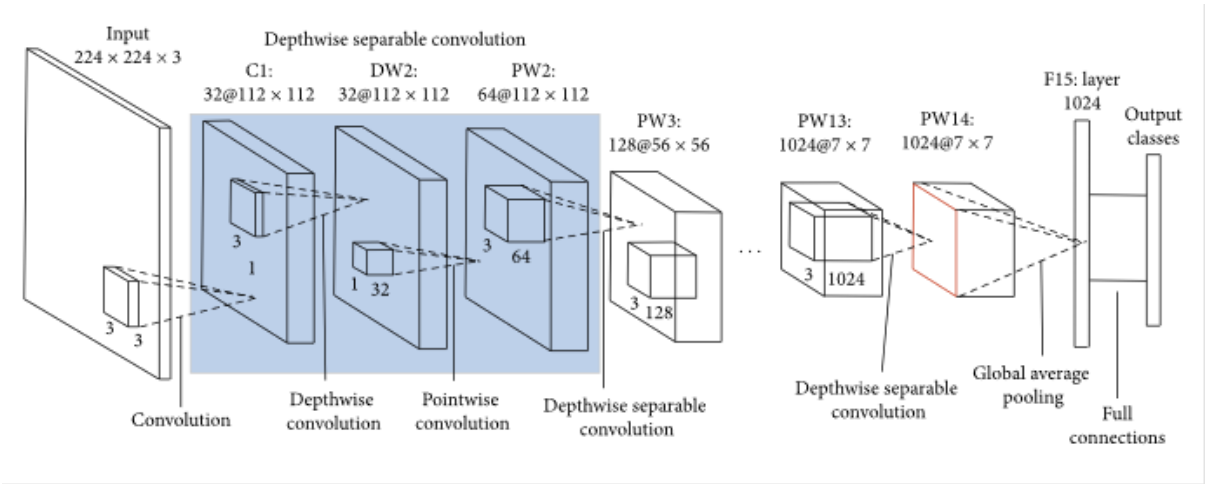
Release Name	Value it adds	Scope	Status	Target date	Challenges
Project scope	Setup the goal for the project	1. Define the business need. 2. Machine Learning problem space	DONE	27 Dec 2021	1. Dataset 2. Models to execute
Project Proposal	Plan high level project execution	1. Project proposal document	DONE	06 Mar 2022	1.Tools
Dataset collection	Setup train and test data for the Deep learning model	1. Dataset collection 2. Labeling the data in the format needed by the model 3. Building the train, validation and test set 4. Data exploration and Munging	IN PROGRESS	26 Mar 2022	
System Design	Defines the deployment architecture	1. Define the tech stack for the web application 2. Finalize the web-services and responsibilities 3. Web application running on local system	DONE	02 Apr 2022	1. Accuracy for ground truth and prediction
Model 1	Shape enhanced accessory transformation	1. Wrap the accessory on person	DONE	10 Apr 2022	1. Locating the landmark with mobilenet 2. Gaze
Model 2	Pose-guided Try-On	1. Rendered accessory on person	DONE	16 Apr 2022	1. Visibility of both the earring for a gazed face 2. Aspect ratio of accessory
Fine Tuning models 1 & 2	Enhanced experience	1. Hyper parameter tuning 2. Handling edge cases	DONE	23 Apr 2022	
MVP - Web application	Building the narration story	1. Data Story 2. Presentation 3. Demo	DONE	30 Apr 2022	

Week-1(Till 2nd April 2022)

- MobileNet:

MobileNet model is based on depth wise separable convolutions. Depth wise separable convolution is made up of two layers: depthwise convolutions and pointwise convolutions. We use depthwise convolutions to apply a single filter per each input channel (input depth). Pointwise convolution, a simple 1×1 convolution, is then used to create a linear combination of the output of the depthwise layer. MobileNets use both batchnorm and ReLU nonlinearities for both layers.

A standard convolution both filters and combines inputs into a new set of outputs in one step. The depthwise separable convolution splits this into two layers, a separate layer for filtering and a separate layer for combining. This factorization has the effect of drastically reducing computation and model size. MobileNet uses 3 × 3 depthwise separable convolutions which uses between 8 to 9 times less computation than standard convolutions at only a small reduction in accuracy. The following figure details the architecture of MobileNet.



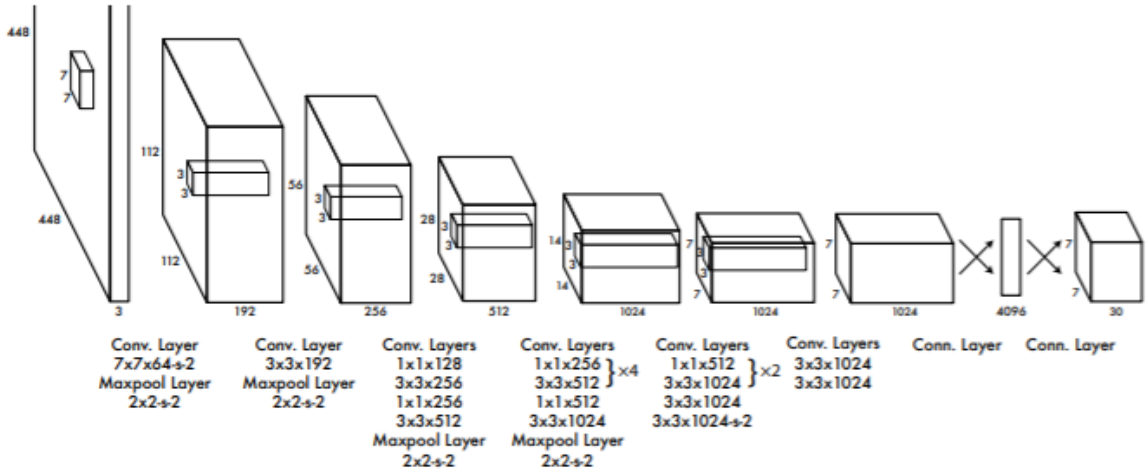
Challenges:

Accuracy for the facial key points and predicted landmark by the model was preety low, which triggered us to try for other models to achive better accuracy.

Week-2(Till 10th April 2022)

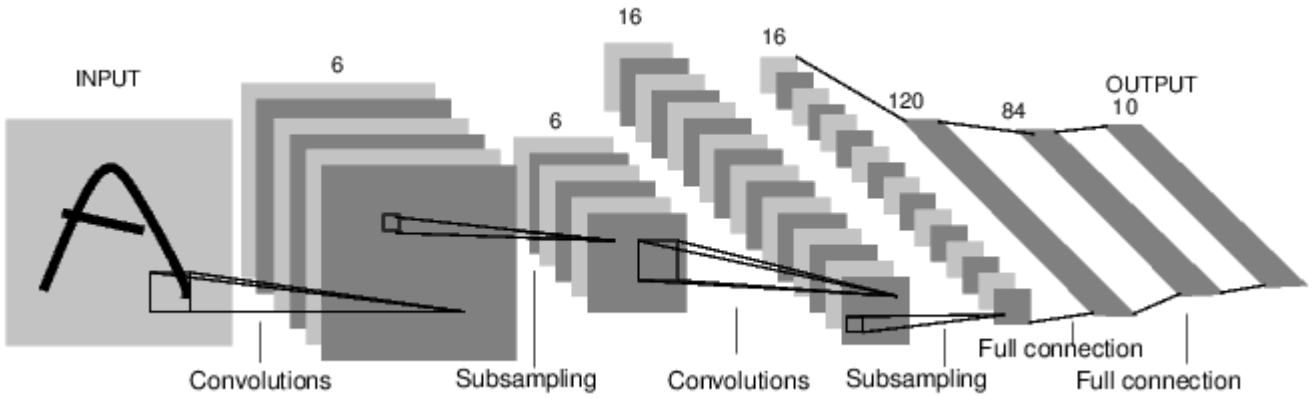
YOLO:

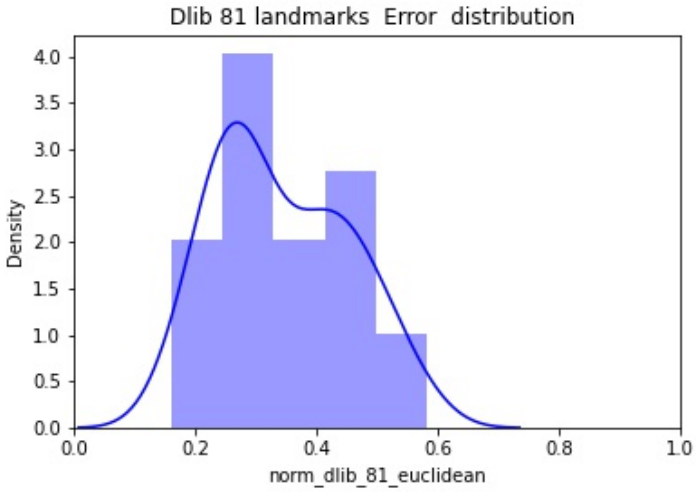
YOLO algorithm uses a completely different approach. The algorithm applies a single neural network to the entire full image. Then this network divides that image into regions which provides the bounding boxes and predicts probabilities for each region. These generated bounding boxes are weighted by the predicted probabilities.



Dlib:

Dlib is **a toolkit for making real world machine learning and data analysis applications in C++**. While the library is originally written in C++, it has good, easy to use Python bindings. I have majorly used dlib for face detection and facial landmark detection. **Shape predictors**, also called **landmark predictors**, are used to predict key *(x, y)*-coordinates of a given "shape". The most common, well-known shape predictor is **dlib's facial landmark predictor** used to localize individual facial structures.





Work Scope:

Once facial key points are accurately predicted, Wrap the accessory on person

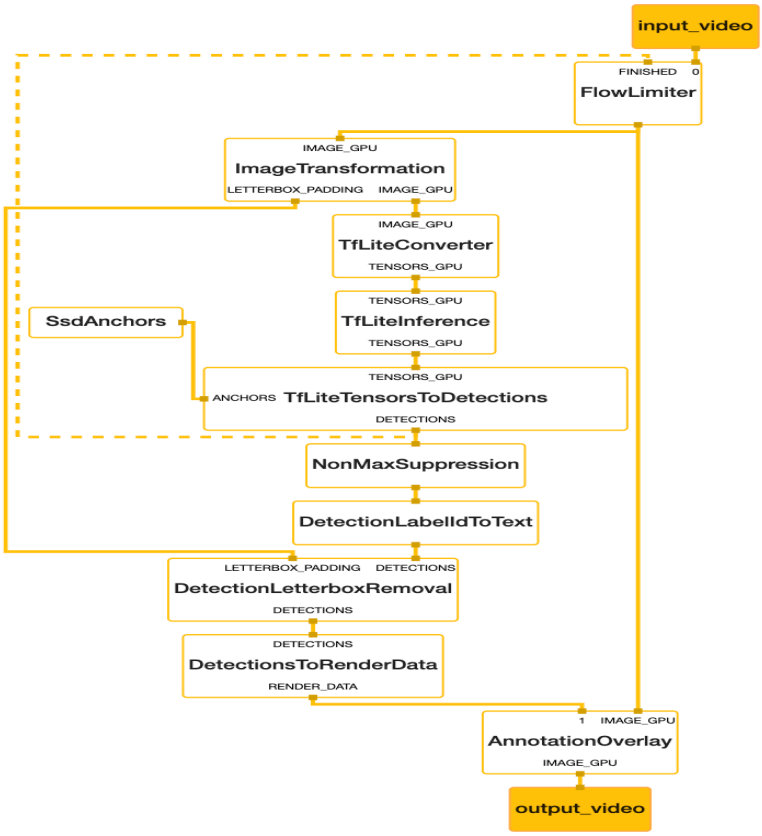
Challenges:

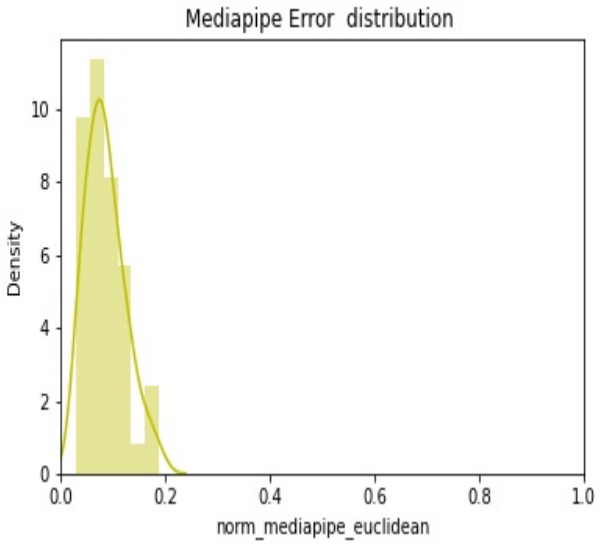
Able to place the accessory correctly on the keypoints for the front facing images only, Gaze/Aspect ratio were not taken into consideration properly, which acts as the driver for searching a better model.

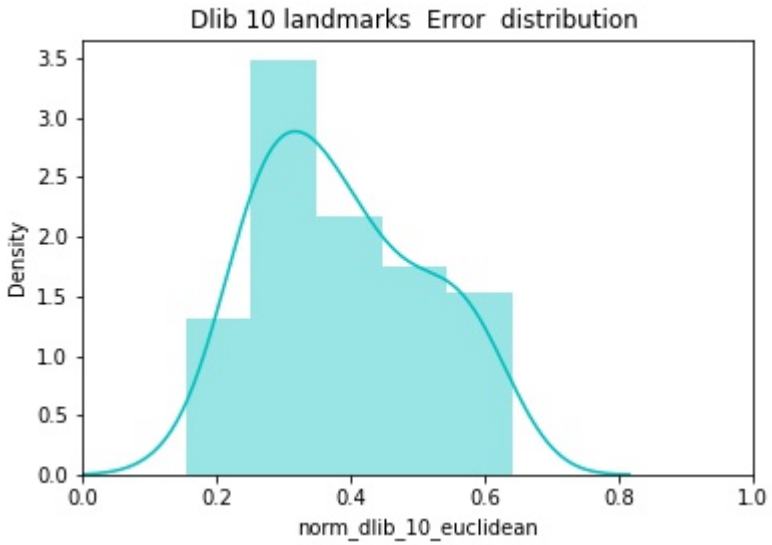
Week-3(Till 16th April 2022)

Mediapipe:

MediaPipe Face Detection is **an ultrafast face detection solution that comes with 6 landmarks and multi-face support**. It is based on BlazeFace, a lightweight and well-performing face detector tailored for mobile GPU inference. MediaPipe is a Framework for building machine learning pipelines for processing time-series data like video, audio, etc. This cross-platform Framework works in Desktop/Server, Android, iOS, and embedded devices like Raspberry Pi and Jetson Nano. MediaPipe Toolkit comprises the **Framework** and the **Solutions.** The following diagram shows the components of the MediaPipe Toolkit.


input_video
FINISHED
0
FlowLimiter
IMAGE_GPU
ImageTransformation
LETTERBOX_PADDING
IMAGE_GPU
IMAGE_GPU
TfLiteConverter
TENSORS_GPU
TENSORS_GPU
SsdAnchors
TfLiteInference
TENSORS_GPU
TENSORS_GPU
ANCHORS
TfLiteTensorsToDetections
DETECTIONS
NonMaxSuppression
DetectionLabelIdToText
LETTERBOX_PADDING
DETECTIONS
DetectionLetterboxRemoval
DETECTIONS
DETECTIONS
DetectionsToRenderData
RENDER_DATA
1
IMAGE_GPU
AnnotationOverlay
IMAGE_GPU
output_video


Mediapipe Error distribution
Density
0
2
4
6
8
10
0.0
0.2
0.4
0.6
0.8
1.0
norm_mediapipe_euclidean



Work Scope:

Rendered accessory on person

Challenges:

Though the accuracy in predicting the keypoints was increased as compared to previous model but ear's midpoint was been predicted instead of earlobes, which triggered us for the better placement of the jewellary.

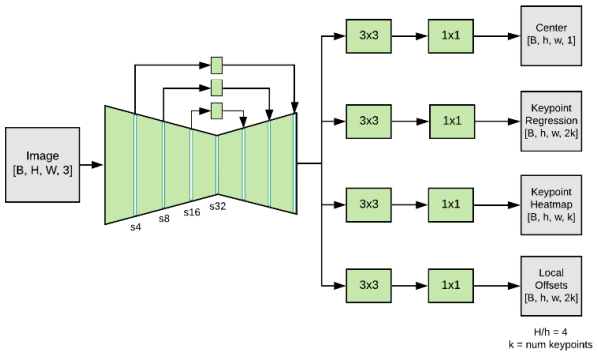
Week-4(Till 23rd April 2022)

- Movenet (solution of the Gaze)

MoveNet is a bottom-up estimation model, using heatmaps to accurately localize human keypoints. The architecture consists of two components: a feature extractor and a set of prediction heads. The prediction scheme loosely follows CenterNet, with notable changes that improve both speed and accuracy. All models are trained using the TensorFlow Object Detection API.

The feature extractor in MoveNet is MobileNetV2 with an attached feature pyramid network (FPN), which allows for a high resolution (output stride 4), semantically rich feature map output. There are four prediction heads attached to the feature extractor, responsible for densely predicting a:

- Person center heatmap: predicts the geometric center of person instances
- Keypoint regression field: predicts full set of keypoints for a person, used for grouping keypoints into instances
- Person keypoint heatmap: predicts the location of all keypoints, independent of person instances
- 2D per-keypoint offset field: predicts local offsets from each output feature map pixel to the precise sub-pixel location of each keypoint



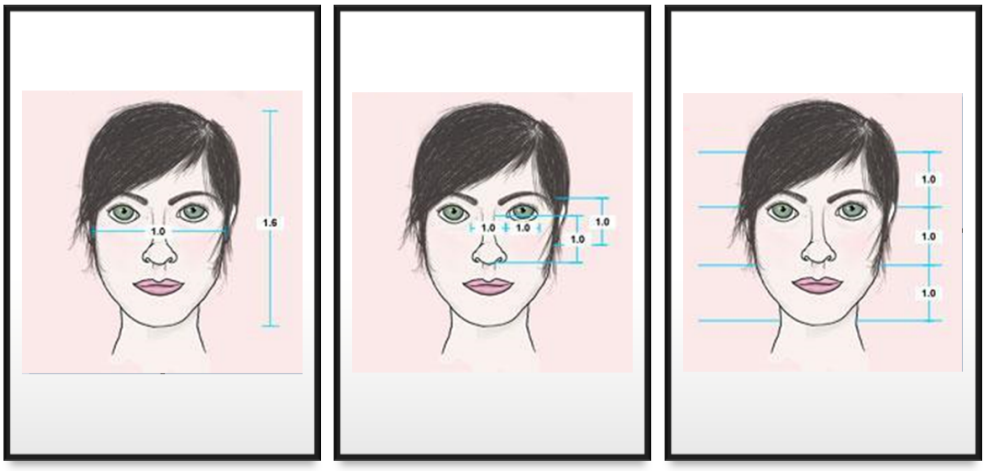
Aspect Ratio:

User's proximity to the camera – near or far, impacts the size of the user face and distance between the facial key-points. After finalizing the correct models for placing the jewelry on the key-points, we realized that the image of the jewelry from the catalog could not be placed directly and needed further processing.

The jewelry size needs to be adjusted before placement on the face accordingly. This aspect was addressed by using the established ideas of golden ratio for the human anatomy.

For an ideal face, **the length of an ear is equal to the length the nose**, and the width of an eye is equal to the distance between the eyes**.** By finding the nose/length of the user, jewelry scaling has been done.

The proportions shown in the following pictures as studied by researchers have been explored to come up with scaling factors for jewelry resizing.



Chapter 4 Project planning

4.1 FutureProposition:

• Test machine learning on real data: When Model has identified a couple of keypoints like Neck, wrist, fingers it would be a good idea to redo this project to determine the suitability of algorithms. So, necklace, bangles, and Rings can be accomodated as well.

• Try out other machine learning platforms: Our model is based on mobilenet, Dlib, Mediapipeline and OpenCV, so in order to increse the picture quality with the ornament, Image generation using GAN can be tested.

• Set up sophisticated data-driven experimental structures: to collect clean and useful data is a challenge, even more if the results have to be very accurate. To use big chunks of data, processes have to be set up, to ensure high quality and quantity of data, access to it and the ability to draw conclusions from it. To implement real-time analytics there are software solutions such as Spark, Hadoop or similar software offered by Amazon Web Services, Microsoft Azure, Google cloud platform.

- Empowering the existing Model: Gaze is not handled effectively in the model. That can be considered for future.

Chapter 5 Conclusion:

In this work, we proposed a novel adaptive content generating and preserving network, which aims at generating photo-realistic try-on results while preserving both the characteristics of jewellary and details of the human identity (posture, body parts). We presented three carefully designed modules, i.e., Mask Generation, Augmentation, and Content Fusion. We evaluated our WearIT model on the realtime videos/images dataset with three levels of try-on difficulties. The results clearly show the great superiority of WearIT over the state-of-the-art methods in terms of quantitative metrics, visual quality, and user study.

Acknowledgements:

We would like to acknowledge Talent Sprint and especially our mentor Mr. Jaley Dholakiya for providing us the necessary guidance and valuable support throughout this capstone project. We value the assistance of IISC professors, learning from their knowledge helped us to become passionate about our project.

Team members' names

- Deepa Raghunathan
- Gargi Golwalkar
- Girish Hariharasubramani
- Sai Divya Ravali
- Som Kanjilal
- Subhendu Kumar Mohapatra
- Urmila Singh

REFERENCES

- https://pyimagesearch.com/2019/12/16/training-a-custom-dlib-shape-predictor/
- https://pyimagesearch.com/2019/12/23/tuning-dlib-shape-predictor-hyperparameters-to-balance-speed-accuracy-and-model-size/
- https://medium.com/analytics-vidhya/image-classification-with-mobilenet-cc6fbb2cd470
- https://towardsdatascience.com/review-mobilenetv1-depthwise-separable-convolution-light-weight-model-a382df364b69
- https://www.oprah.com/oprahshow/measuring-facial-perfection-the-golden-ratio#:~:text=Schmid%20measures%20the%20length%20and,B.
- https://www.goldennumber.net/face/
- https://www.outlookindia.com/business/indian-gold-jewellery-consumers-hold-back-buying-on-expectations-of-lower-price-news-193753
- https://google.github.io/mediapipe/solutions/hands.html
- https://en.wikipedia.org/wiki/Euclidean_distance
- https://towardsdatascience.com/9-distance-measures-in-data-science-918109d069fa
- https://www.analyticsvidhya.com/blog/2021/06/implementation-of-yolov3-simplified/#:~:text=YOLO%20is%20a%20Deep%20Learning,detection%20used%20in%20real%2Dtime.