MACHINE LEARNING

FINAL PROJECT REPORT

TEAM-18

MANSI SOMAYAJULA

NIKHIL DAMERA

PRUDHVI KUMAR REDDY KETHANAPALLY

SARANYAKUMAR CHIGURAPATI

e-FASHION (FASHION COLLABORATING WITH COMPUTER VISION)

We designed a predictive model which works on Fashion items using computer vision.

We used Mask RCNN to design the system.

Implemented in Google Colab.

The following are the steps which we followed to design this system.

Step 1: Importing libraries which are required to carry basic functionalities.

Step 2: The data which is used for this project is in Google Drive. So in this step, we are mounting the drive to Colab.

Step 3: Now, specifying paths for the following variables.

- TRAIN_IMAGE_DIR
- DATA_DIR
- ROOT_DIR

IMPORTING DATA

Step 4: With the help of pandas library, we are importing the data.

Step 5: Extracting the meta data of image from the Label.json file. This produces the output of all the labels which are present in the Label.json file.

Once we obtain the label names, we are then assigning the attribute names. Here we assigned ‘Id’ and ‘Label’ as the column names and added the labels to a data frame.

	Id	Labels
0	0	shirt, blouse
1	1	top, t-shirt, sweatshirt
2	2	sweater
3	3	cardigan
4	4	jacket

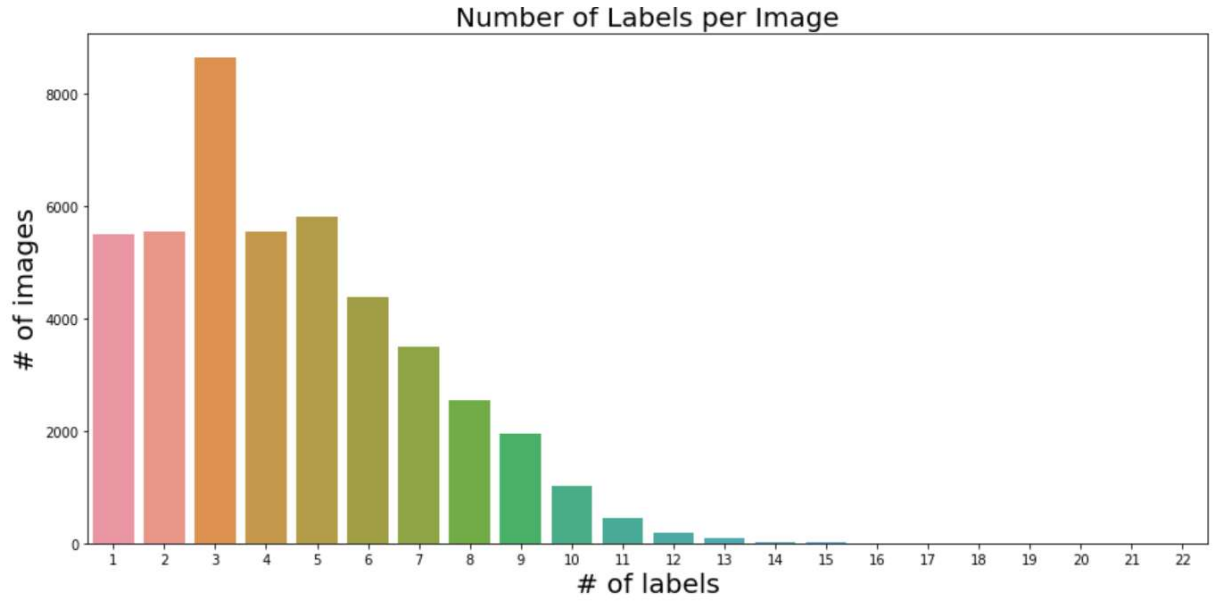
Step 6: Adding each segment to a data frame.

```
Total segments:  201824
```

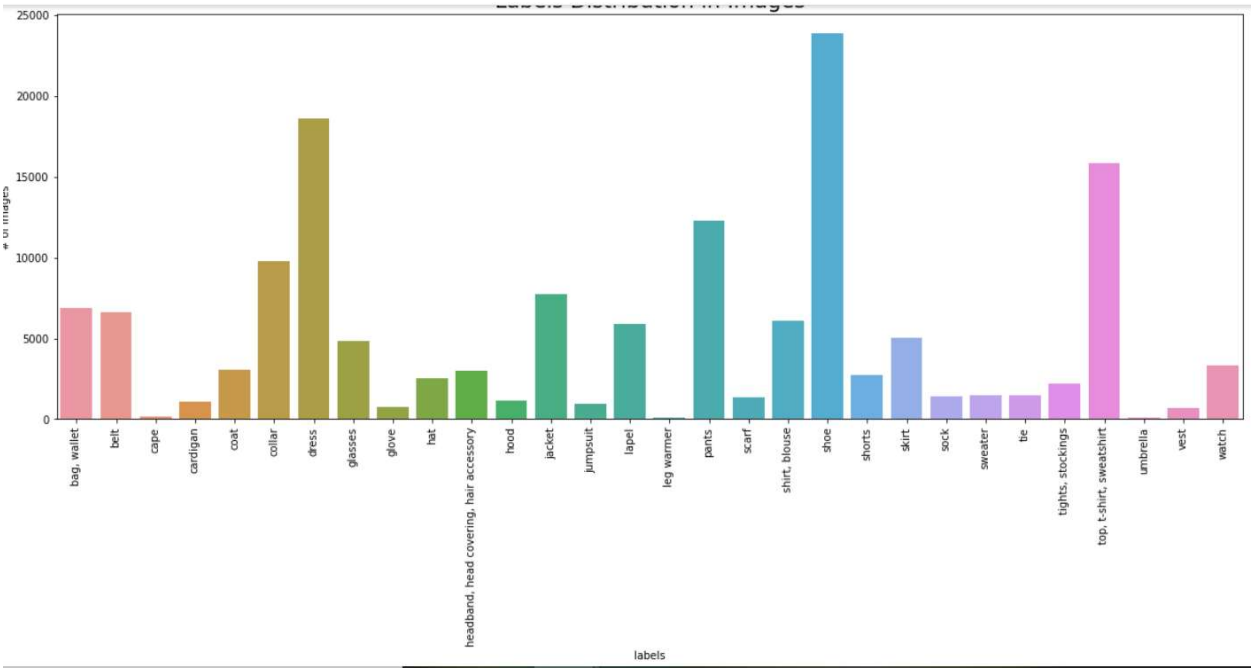
	ImageId	EncodedPixels	Height	Width	ClassId	CategoryId
10	0000fe7c9191fba733c8a69cfaf962b7.jpg	1343707 9 1346138 27 1348569 44 1351000 62 135...	2448	2448	1	1
24	00048c3a2fb9c29340473c4cfc06424a.jpg	257702 10 258670 26 259640 36 260572 6 260610 ...	975	650	1	1
31	0006ea84499fd9a06fefbdf47a5eb4c0.jpg	73327 6 74328 18 75330 37 76335 63 77340 88 78...	1000	667	1	1
37	000775b6b5e27b011dc8bba2d7b85211.jpg	309670 11 311259 33 312848 55 314440 72 316033...	1600	1067	1	1
54	000b3ec2c6eaffb491a5abb72c2e3e26.jpg	458445 17 460731 51 463017 85 465303 115 46759...	2310	1536	1	1

Step 7: Rows with the same image are grouped together because the subsequent operations perform at an image level

Step 8: Creating a data frame for images. And printing the labels per image.



Step 9: Creating a bar graph which contains the details about labels distribution of images.



PARTITION OF DATA TO TRAINING SET AND TESTING SET.

Step 10: We are using Mask R-CNN with COCO pretrained weights to the task.

MASK R-CNN IMPLEMENTATION

We are cloning into Mask R-CNN.

CONFIGURAITONS

We are using tensorflow of version 1.14.0 and keras version 2.2.5.

Step 11: Mask R-CNN has a load of hyperparameters. Displaying the configurations of the images such as IMAGE_META_SIZE, IMAGE_SHAPE, IMAGE_SIZE etc.

Step 12: Customizing function which is used to resize the image.

VISUALIZATION

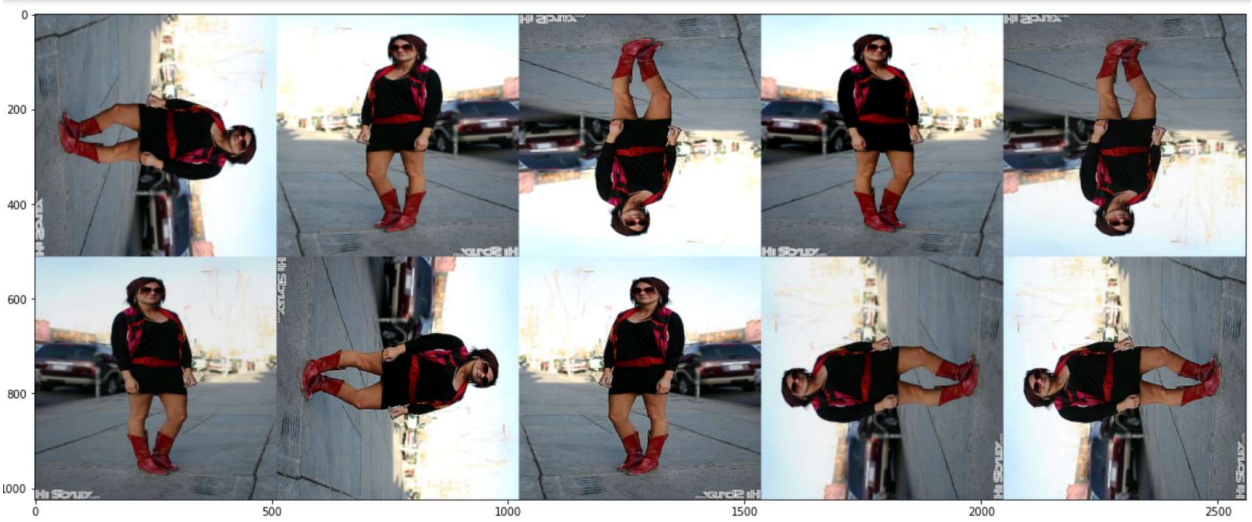
Step 13: Visualizing random images from the images data frame. We took a sample image and got a data frame which identified the following items: Vest, dress, glasses, belt, pair of shoes, lapel.

PREPARING THE DATA

Step 14: Preparing the data for training and validation.

SAMPLE OUTPUT

Step 15: Augmenting the random image and we represented it in a data grid.



TRAINING THE MODEL

Step 16: Initiating Mask R-CNN training. Here we declared the running rate as 0.0003. After training the model with 10 epoch's,we obtained the loss value, wall time value.

```
Epoch 7/10
500/500 [==============================] - 622s 1s/step - loss: 8.5160 - rpn_class_los
s: 0.0178 - rpn_bbox_loss: 3.3415 - mrcnn_class_loss: 0.1090 - mrcnn_bbox_loss: 2.3099
- mrcnn_mask_loss: 2.7379 - val_loss: 8.4445 - val_rpn_class_loss: 0.0221 - val_rpn_bbo
x_loss: 3.2934 - val_mrcnn_class_loss: 0.1387 - val_mrcnn_bbox_loss: 2.3302 - val_mrcnn
_mask_loss: 2.6601
Epoch 8/10
500/500 [==============================] - 588s 1s/step - loss: 8.0878 - rpn_class_los
s: 0.0157 - rpn_bbox_loss: 3.2621 - mrcnn_class_loss: 0.1107 - mrcnn_bbox_loss: 2.1475
- mrcnn_mask_loss: 2.5518 - val_loss: 7.9278 - val_rpn_class_loss: 0.0187 - val_rpn_bbo
x_loss: 3.3723 - val_mrcnn_class_loss: 0.0989 - val_mrcnn_bbox_loss: 2.1074 - val_mrcnn
_mask_loss: 2.3305
Epoch 9/10
500/500 [==============================] - 611s 1s/step - loss: 7.6927 - rpn_class_los
s: 0.0134 - rpn_bbox_loss: 3.1940 - mrcnn_class_loss: 0.1017 - mrcnn_bbox_loss: 2.0152
- mrcnn_mask_loss: 2.3684 - val_loss: 7.2501 - val_rpn_class_loss: 0.0139 - val_rpn_bbo
x_loss: 3.1571 - val_mrcnn_class_loss: 0.0630 - val_mrcnn_bbox_loss: 1.8922 - val_mrcnn
_mask_loss: 2.1240
Epoch 10/10
500/500 [==============================] - 638s 1s/step - loss: 7.5171 - rpn_class_los
s: 0.0125 - rpn_bbox_loss: 3.0757 - mrcnn_class_loss: 0.1197 - mrcnn_bbox_loss: 1.9528
- mrcnn_mask_loss: 2.3564 - val_loss: 7.3177 - val_rpn_class_loss: 0.0171 - val_rpn_bbo
x_loss: 3.0766 - val_mrcnn_class_loss: 0.1139 - val_mrcnn_bbox_loss: 1.8433 - val_mrcnn
_mask_loss: 2.2667
CPU times: user 44min 6s, sys: 4min 47s, total: 48min 54s
Wall time: 1h 45min 11s
```

Step 17: We are finding the best epoch value and the loss associated with it

Best epoch: 15

Valid loss: 6.721006140708924

Now we need to load the trained model which will be saved in .h5 extension file and test with a random image

Step 18: Testing the random image by making the similar data changes which were done during the training step.

The original Image is shown below:

The masked image is shown below:

With the above image we can deduce a fact that the machine is able to predict and mask an image up to a certain amount of accuracy and the labels mentioned as “top, t-shirt, sweatshirt”.